Sunhayato

SR8C15CP マイコンIC

スタートアップガイド

E8エミュレータ導入編 1

2004年12月15日発行
REV. 1. 00
SG043101

サンハヤト株式会社
本社 〒170-0005 東京都豊島区南大塚3-40-1
☎ 03-3984-7791　FAX. 03-3971-0535
http://www.sunhayato.co.jp

お願いとご注意

＜本資料について＞

- 本資料は、電子工作や電子回路、パーソナルコンピュータの操作について一般的な知識をお持ちの方を対象にしています。
- 本資料を元に操作するには、株式会社 ルネサス テクノロジ社製 R8C/Tiny マイコンについての知識や開発環境などが必要です。

＜安全設計に関するお願い＞

- 一般的に半導体を使用した製品は、誤動作したり故障することがあります。半導体の誤動作や故障の結果として事故や損害などを生じさせないように考慮した安全設計をご購入者の責任で行ってください。

＜本資料のご利用にあたって＞

- この取扱説明書に掲載している内容は、お客様が用途に応じた適切な製品をご購入頂くことを目的としています。その使用により当社及び第三者の知的財産権その他の権利に対する保証、又は実施権の許諾を意味するものではありません。また、権利の侵害に関して当社は責任を負いません。
- 本資料に記載した情報を流用する場合は、お客様のシステム全体で充分評価し適用可能かご判断願います。当社では適用可能判断についての責任は負いません。
- 本資料に記載してある内容は、一般的な電子機器（学習教材、事務機器、計測機器、パーソナル機器、コンピュータ機器など）に使用されることを目的としています。高い品質や信頼性が要求され、故障や誤作動が直接人命を脅かしたり人体に危害を及ぼす恐れのある、医療、航空宇宙、原子力制御、運輸、移動体、各種安全装置などの機器への使用は意図も保証もしておりません。
- この取扱説明書の一部、又は全部を当社の承諾なしで、いかなる形でも転載又は複製されることは堅くお断りします。
- 全ての情報は本資料発行時点のものであり、当社は予告なしに本資料に記載した内容を変更することがあります。
- この資料の内容は慎重に制作しておりますが、万一記述誤りによってお客様に損害が生じても当社はその責任を負いません。
- 本資料に関してのお問合せ、その他お気付きの点がございましたら、当社までお問合せください。
- 本資料に関する最新の情報はサンハヤト株式会社ホームページ（http://www.sunhayato.co.jp/）に掲載しております。

このマニュアルについて

本スタートアップガイドでは、SR8C15CP マイコン IC のプログラム開発を、ルネサステクノロジ社の E8 エミュレータおよび HEW（High-Performance Embedded Workshop）を使用して行った場合の手順、設定方法などを説明したものです。E8 エミュレータ、HEW の詳細な使用方法や注意事項につきましては、必ずルネサステクノロジ社発行のマニュアルを参照してください。

このスタートアップガイドでは、以下の OS、ツールのバージョンで動作したものとして説明しています。

ホスト PC の OS：Windows 2000
E8 エミュレータ：V.1.00.01
統合化開発環境：HEW　V.3.01.04.001
C コンパイラ：NC8C　V.5.30 Release 1

E8 エミュレータ、HEW のバージョンは、E8 同梱の CD の Je8rm.txt で確認できます。また HEW は、HEW の「ヘルプ」メニューの「High-Performance Embedded Workshop のバージョン情報」でも確認できます。

Microsoft®, Windows®は米国 Microsoft 社の米国およびその他の国における商標登録です。
IBM は International Business Machine Corporation の商標登録です。
その他、記載されている製品名は各社の商標または商標登録です。

1 SR8C15CP

1-1 SR8C15CP の特長

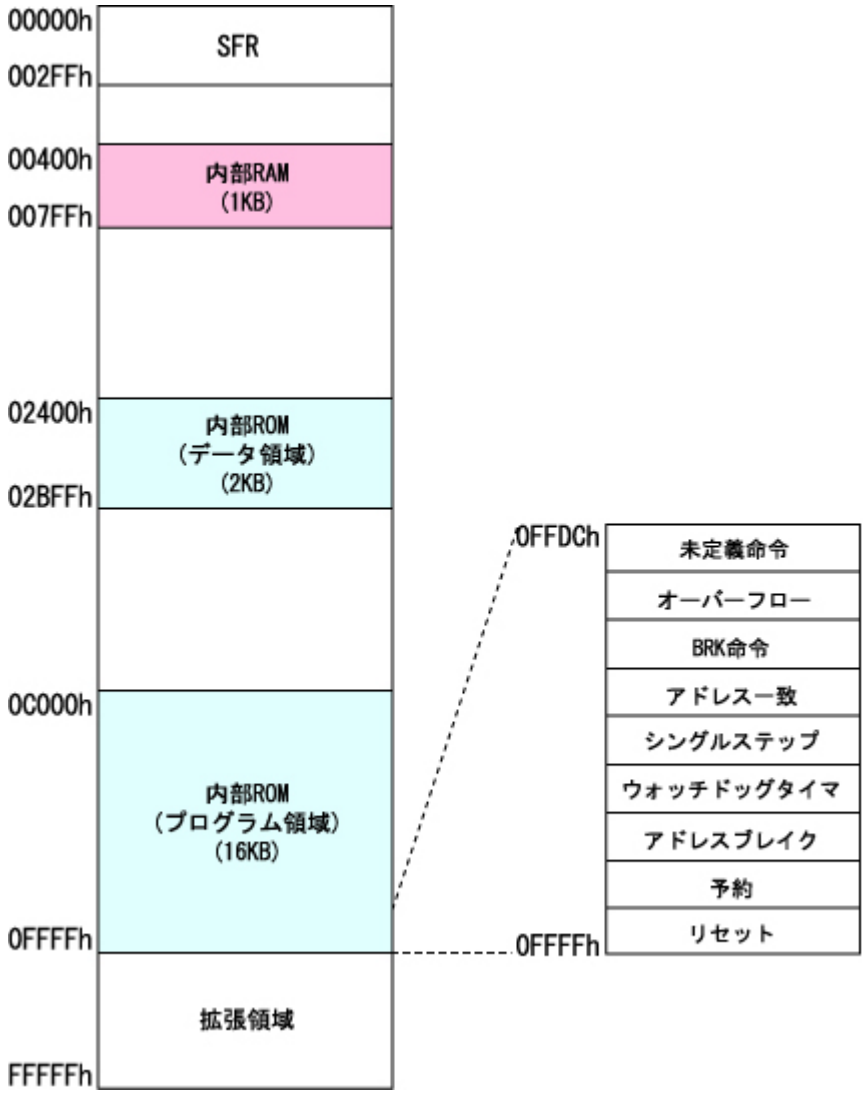
サンハヤトオリジナルマイコン SR8C15CP は、 ルネサステクノロジ社マイクロコンピュータ R8C/Tiny シリーズ 15 グループ　R5F21154SP を使いやすい 20 ピンプラスチックモールド DIP に収めたものです。
詳細については　ルネサステクノロジ社「R8C/14、R8C/15 グループ　ハードウェアマニュアル」を参照してください。

表 1-1 SR8C15CP 概要

CPU コア	ルネサス テクノロジ社製 16 ビット CPU コア：R8C CPU
最大動作周波数／電源電圧	20MHz / 3.0V ～ 5.5V 10MHz / 2.7V ～ 5.5V
動作温度範囲	－20～85℃
フラッシュメモリ（プログラム領域）	16k バイト
フラッシュメモリ（データ領域）	1k バイト×2 ブロック
RAM	1k バイト
多機能タイマ	16 ビットタイマ：1 チャネル(インプットキャプチャ／アウトプットコンペア機能) 8 ビットタイマ：2 チャネル
ウォッチドッグタイマ	15 ビット 1 チャネル(H/W リセット対応)
シリアルインタフェース	クロック同期／UART：1 チャネル チップセレクト付同期式シリアル I/O（SSU）
A/D コンバータ	10 ビット A/D コンバータ 1 回路、4 チャネル
汎用ポート	プログラマブル入出力ポート：13 本 入力専用ポート：2 本
LED 駆動ポート	入出力 4 本
プルアップ抵抗	入力ポートを除く全ポートに 4 ビット単位で設定可能
パワーオンリセット回路	外部のリセット回路なしでマイコンをリセットする機能
電圧検出回路	VCC 端子に入力する電圧を Vdet1 、Vdet2 で監視する回路
発振回路	メインクロック発振回路（メインクロック発振停止検出機能内蔵） オンチップオシレータ（125KHz／8MHz 切り替え可）
パッケージ	2.54mm ピッチ 20 ピン DIP
用途	学習／機能評価用
パッケージに関連する仕様以外は、R5F21154SP と同等です。	

1-2 SR8C15CP のメモリ配置

以下に SR8C15CP のメモリ配置を示します。



SFR : Special Function Register の略で、主に周辺機能のレジスタが配置されています。
内部 ROM（データ領域）: Flash ROM で、プログラム/イレーズが 10,000 回可能です。
内部 ROM（プログラム領域）: Flash ROM で、プログラム/イレーズが 100 回可能です。

図 1-1 SR8C15CP メモリマップ

1-3 SR8C15CP のトライアル基板回路例

以下に SR8C15CP のトライアル基板回路例を示します。E8 エミュレータを接続することを想定しています。また電源は E8 より 5V を供給します。
電源インジケータ用に 1 つ（燈）、テスト用に 2 つ（赤、緑）の LED を付けています。

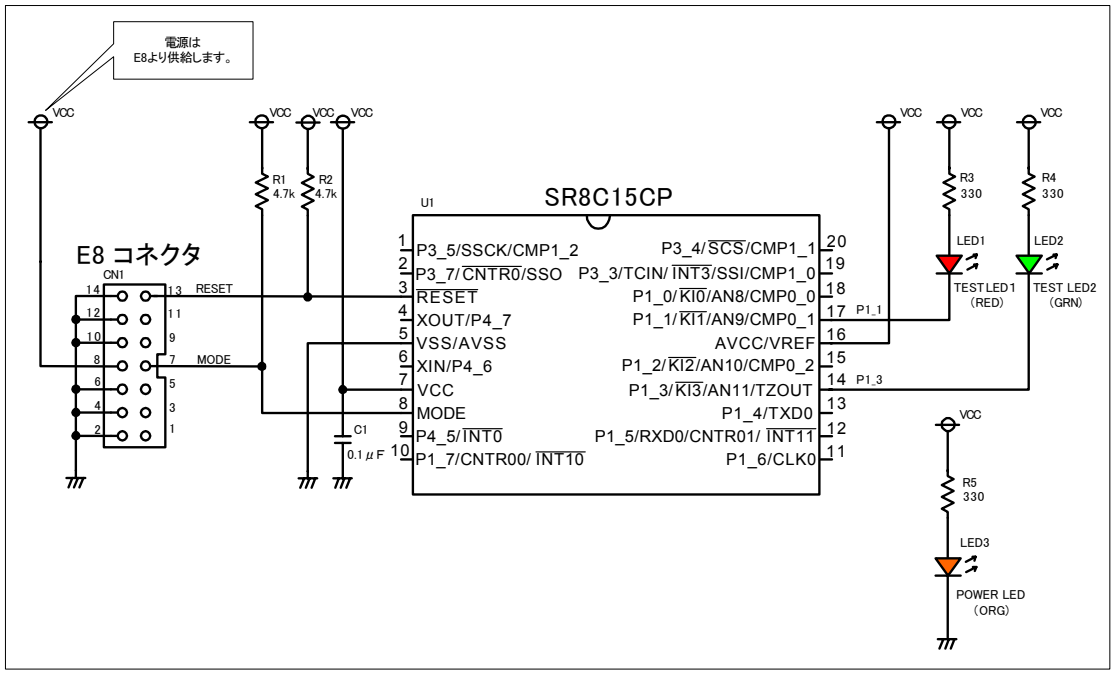


図 1-2 SR8C15CP トライアル基板回路例

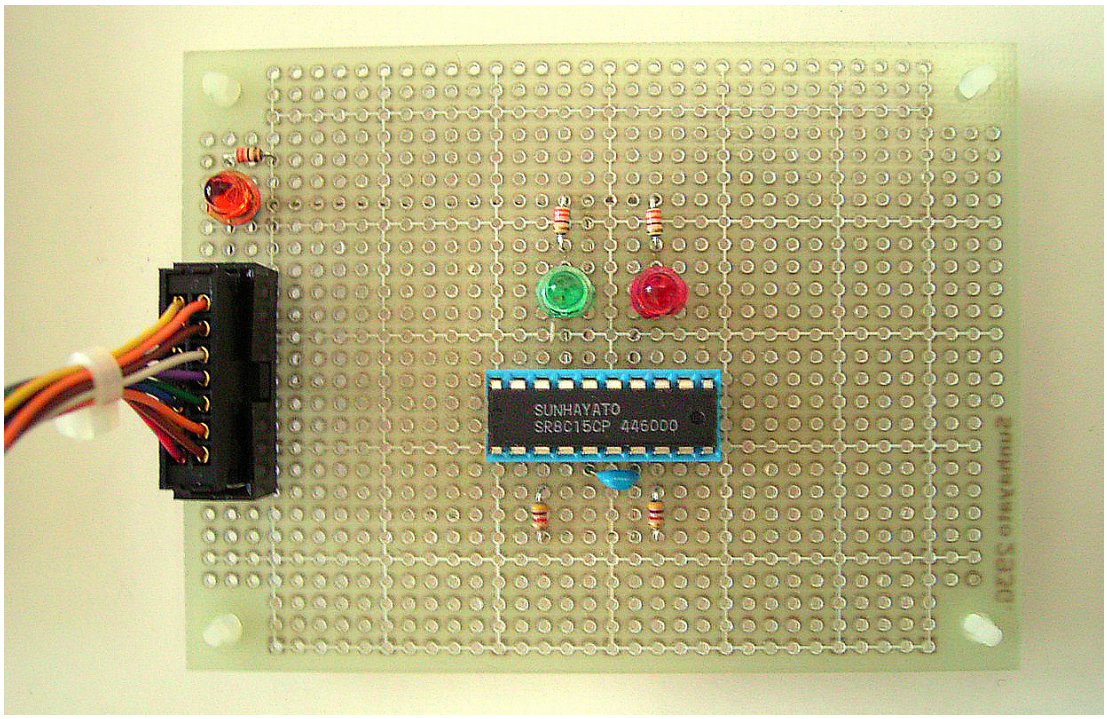


図 1-3 トライアル基板と E8 の接続

2　E8 エミュレータ

2-1 E8 と HEW

E8 はルネサステクノロジ社 R8C/Tiny マイクロコンピュータ専用のオンチップデバッギングエミュレータです。マイクロコンピュータとは MODE 端子（ユーザシステムでは使用しない端子です）1 本のみで通信することができます。ホスト PC 側のアプリケーションとしては HEW（Renesas High-performance Embedded Workshop）を使用することになります。HEW では、デバッグ時に E8 を介してマイクロコンピュータのフラッシュメモリにファームウエアを書き込み、ホスト PC 上のプログラムと通信するリモートデバッガ方式をとりますが、SR8C15CP の場合はファームウエアの書き込み先としてプログラム領域（C000h～C7FFh）に加え、データ領域（2400h～2BFFh）を指定できますので、ターゲットシステムへの影響を極力抑えながらデバッグをすることができます。

E8 では、ホスト PC とは USB ケーブルで接続します。ホスト PC から E8 への電源供給はもちろん、E8 を介してターゲットボードへ電源を供給することもできます。ターゲットへの電源供給は 3V と 5V から選択できます。

HEW と E8 エミュレータを使用した SR8C15CP 開発フロー（イメージ図）を以下に示します。

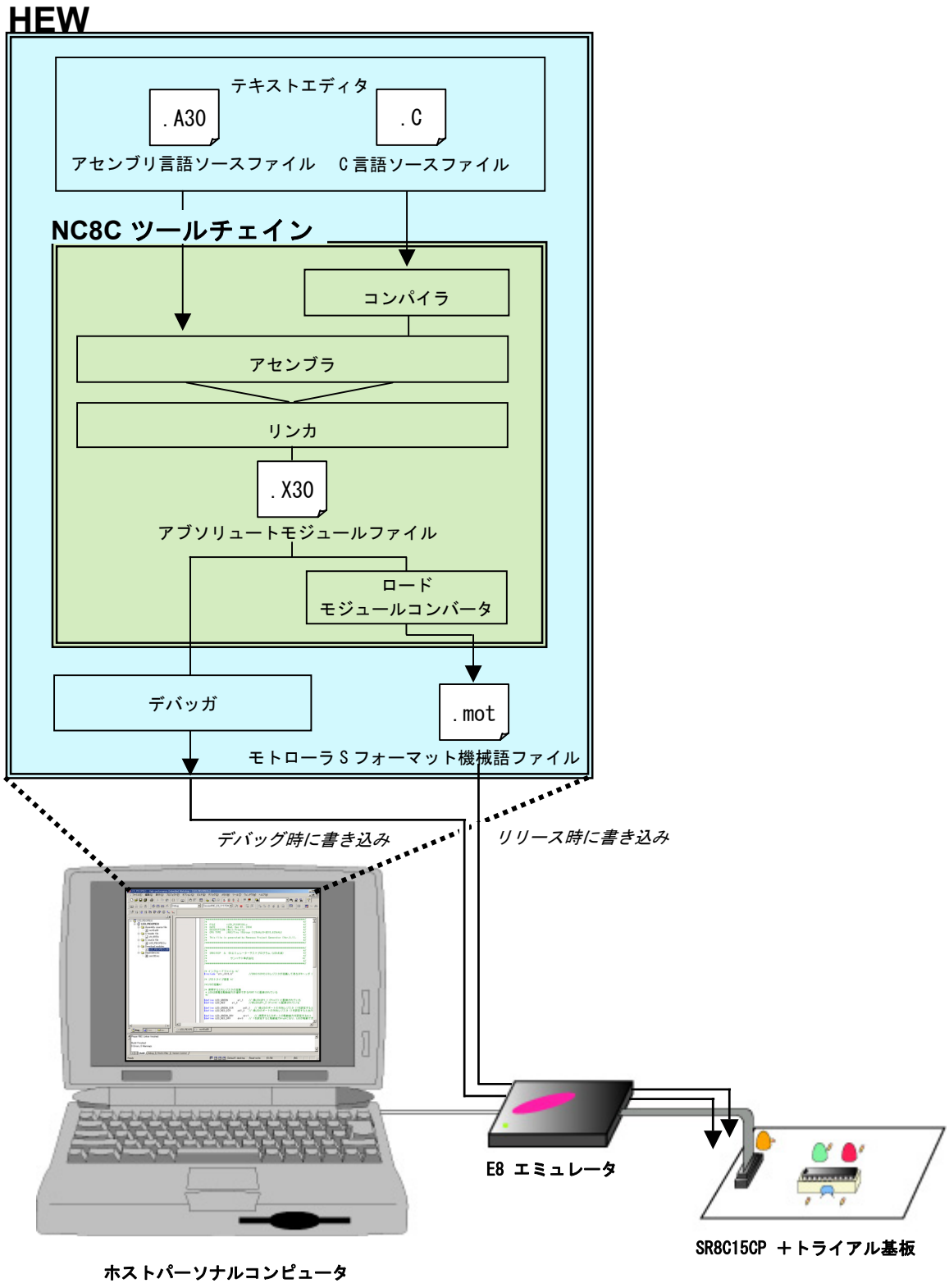


図 2-1 HEW と E8 による開発フロー（イメージ図）

3 サンプルプログラムを動かしてみよう

3-1 HEW＋E8 での開発に必要なもの

・ ホストパーソナルコンピュータ（以下の条件を満たすもの）
IBM PC/AT 互換機（Windows XP, Windows Me, Windows 98SE, Windows 2000）
USB インタフェース（USB 1.1, フルスピード）USB2.0 対応でも接続できます。
CD-ROM ドライブ（E8 同梱 CD からのインストール）

・ SR8C15CP 用トライアル基板（図 1-2 の回路図を参考に製作してください）

3-2 開発環境を整えよう

E8 に付属の CD に収められているプログラム開発に必要なツール類をインストールしてください。また、R8C/Tiny 専用無償コンパイラ NC8C、ルネサス統合化環境 HEW は、ルネサステクノロジ社サイトからもダウンロードできます。

3-3 サンプルプログラムをダウンロードしよう

サンハヤトのサイトよりサンプルプログラム（LED_PICOPICO.lzh）をダウンロードしてください。サンプルプログラムは HEW で開発したものです。Lzh 形式の圧縮ファイルですので、解凍する必要があります。サンハヤトの R8C/Tiny ページ→ http://www.sunhayato.co.jp/tiny/

HEW ではフォルダ名に英数文字と、アンダーバー“_”が使用できます。以降紹介する例では、コピーした LED_PICOPICO.lzh を C:¥HEW3¥Program フォルダへ解凍したこととしています。

図 3-1 サンハヤト R8C/Tiny のページ

3-4 サンプルプログラムの構成

以下にサンプルプログラムの構成を示します。
HEW では、プログラムのソースファイルと、ソースファイルからオブジェクトファイル（マイクロコンピュータが実行可能な機械語ファイル）を生成するまでの手順や設定をまとめて、１つの「プロジェクト」として扱います。
また、「プロジェクト」はさらに「ワークスペース」という大きなグループに属しています。１つのワークスペースに複数のプロジェクトが属している場合もあります。
サンプルプログラムでは LED_PICOPICO というワークスペースに同名のプロジェクトが１つ属しています。

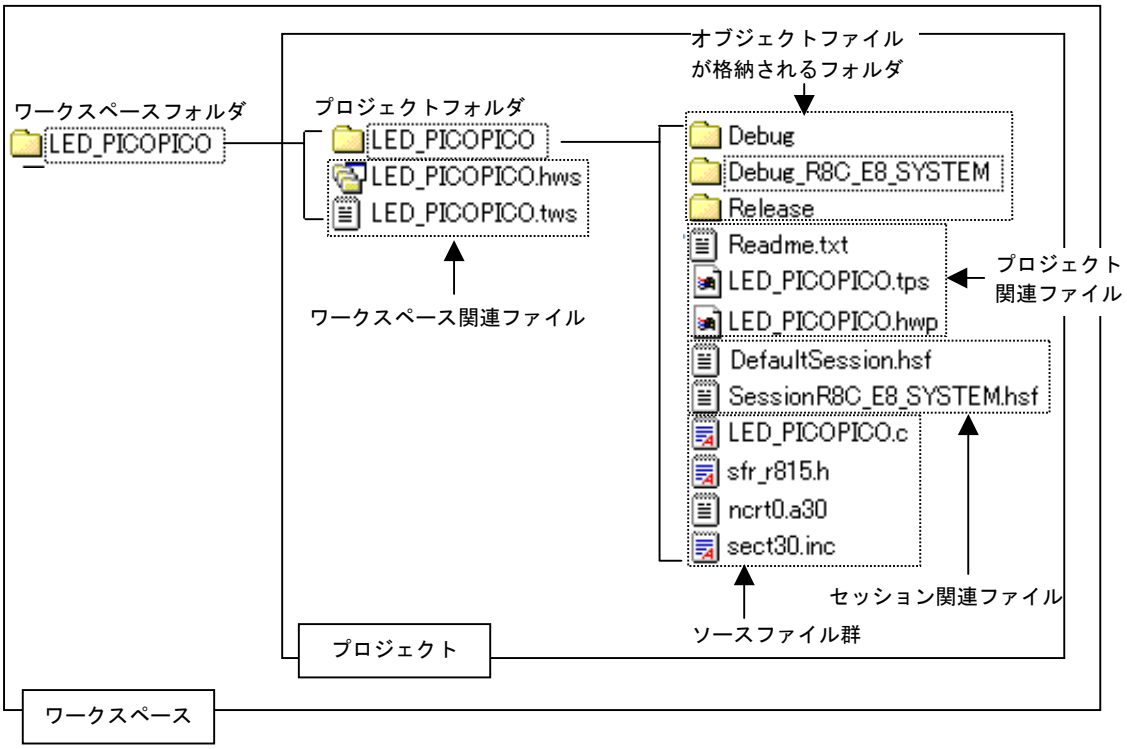


図 3-2 サンプルプログラムの構成

3-5 ワークスペースを開く

HEW を起動します。Windows のスタートメニューより、「プログラム」→「Renesas High-performance Embedded Workshop」の 「High-performance Embedded Workshop」をクリックしてください。HEW が起動します。「別のプロジェクトワークスペースを開く」にチェックを入れ、「OK」をクリックしてください。

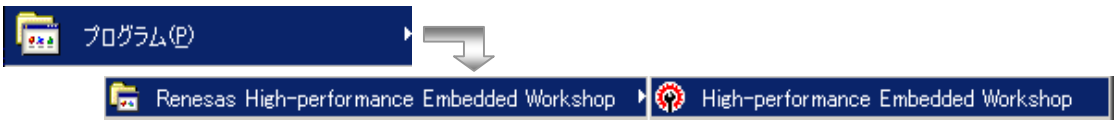


図 3-3 HEW:起動する

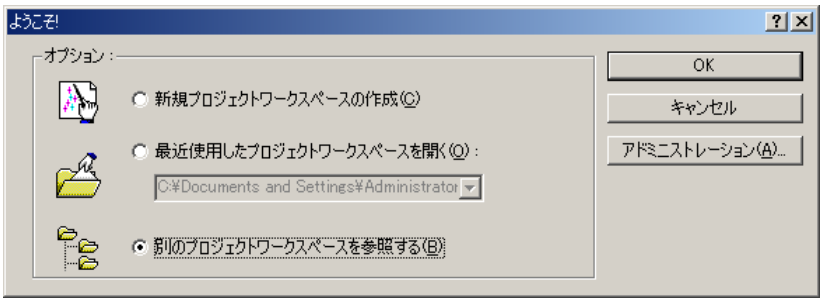


図 3-4 HEW:プロジェクトを開く

コピーした LED_PICOPICO フォルダ内の「LED_PICOPICO.hws」ファイルを選んで「開く」をクリックしてください。LED_PICOPICO ワークスペースは「C:¥R8C_PROGRAM フォルダ」内に作成したものですので、違う構成のフォルダへコピーした場合は以下のようなメッセージが出ます。「はい」を押してください。このメッセージはフォルダを移動したあと初回のワークスペース参照時のみ出ます。(ここでは C:¥HEW3¥Program フォルダへコピーしたこととしています。)

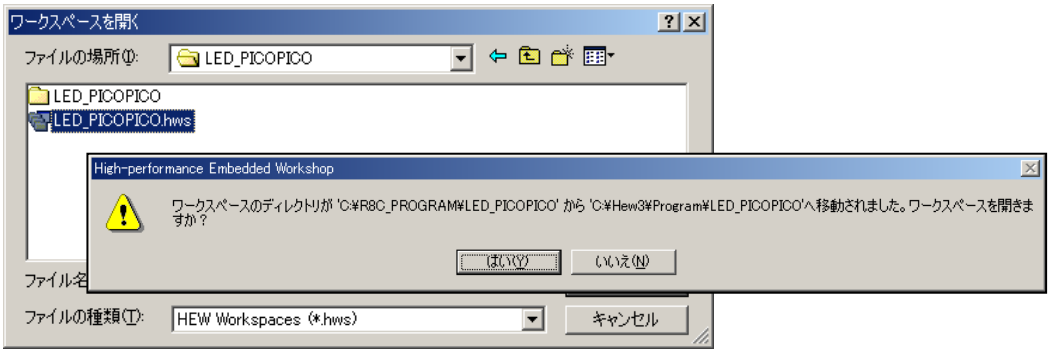


図 3-5 HEW:ワークスペースの移動メッセージ

ワークスペースが開きます。

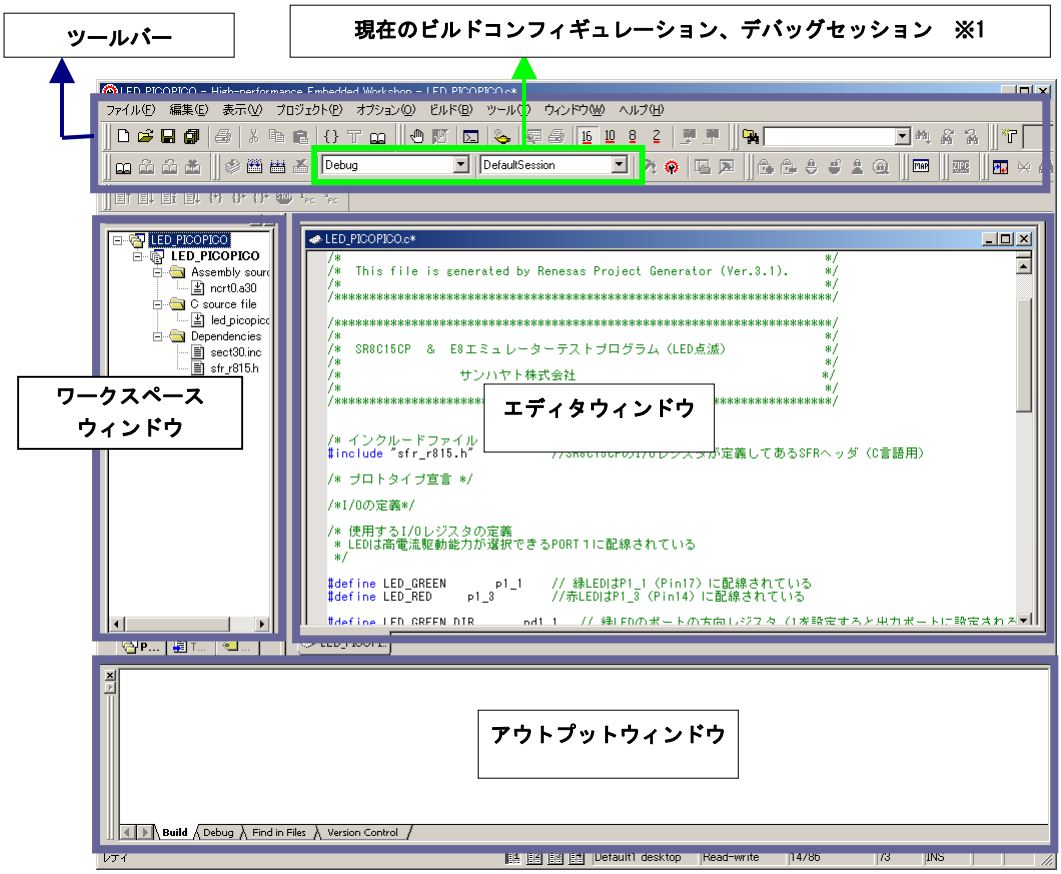


図 3-6 HEW:ワークスペース

※1　ビルドコンフィギュレーション、デバッグセッションについては次章「3-6 ビルドコンフィギュレーション、デバッグセッションについて」を参照してください。

3-6 ビルドコンフィギュレーション、デバッグセッションについて

「ビルドコンフィギュレーション」には、ビルドを行う際のフェーズ(アセンブラ、コンパイラ、リンカ）の構成や、各フェーズでのオプション設定が保存されています。また、「デバッグセッション」には、デバッグを行う際のデバッギングプラットフォーム（E8 エミュレータ等）との接続のための各設定等が保存されています。このコンフィギュレーションを切り替えることによってビルド(コーディング)→デバッグ→最終リリースの各作業を効率良く行うことができます。サンプルプログラムでは各コンフィギュレーションを以下のように設定しています。

＜ビルドコンフィギュレーション＞

Debug: 評価中の設定としています。コンパイラオプションに-g オプション（デバッグ情報をリロケータブルファイル(.x30)に含める）を追加しています。このコンフィギュレーションでビルドした場合、生成された各ファイルは LED_PICOPICO プロジェクトフォルダ内の Debug フォルダに格納されます。

Release: このサンプルでは、特に設定はしていません。このコンフィギュレーションでビルドした場合、生成された各ファイルは LED_PICOPICO プロジェクトフォルダ内の Release フォルダに格納されます。

Debug_R8C_E8_SYSTEM :
このサンプルでは、特に設定はしていません。このコンフィギュレーションでビルドした場合、生成された各ファイルは LED_PICOPICO プロジェクトフォルダ内の Debug_R8C_E8_SYSTEM フォルダに格納されます。

＜デバッグセッション＞

DefaultSession: E8 とは接続せず、ホスト PC はスタンドアロンの状態で作業を行います。

Session_R8C_E8_System: E8 との接続のための設定が登録されています。このセッションに切り替えると E8 との接続ウィザードが開始され、ホスト PC は E8 を介してターゲットシステムと接続されます。

3-7 ビルドする

ツールバーの「ビルド」メニューの「ビルド」をクリックすると、ビルドが開始され、各フェーズでのメッセージがアウトプットウィンドウに表示されます。（ビルドアイコンをクリックしても同じくビルドを開始できます。）

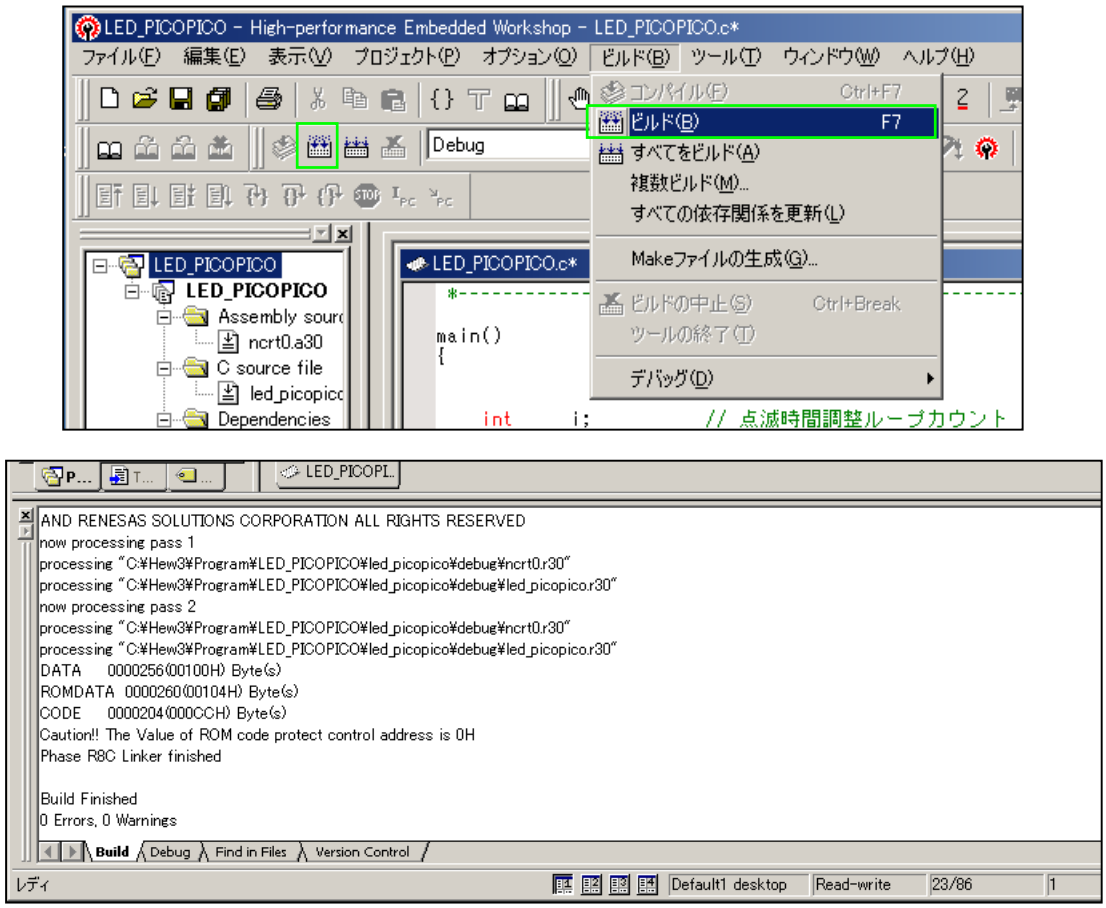


図 3-7 HEW:ビルドする

3-8 E8 と接続する

ホスト PC と E8 を接続します。ホスト PC に E8 を接続し、E8 をターゲットボードに接続してください。
デバッグセッションを「DefaultSession」から「Session_R8C_E8_System」へ切り替えます。

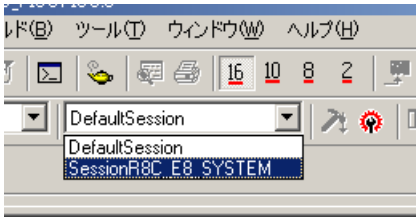


図 3-8 HEW:セッションを切り替える

E8 との接続ウィザードが始まります。「Select Emulator Mode」ウィンドウで以下のように設定してください。

Device: R5F21154
Mode: Download emulator firmware にチェック

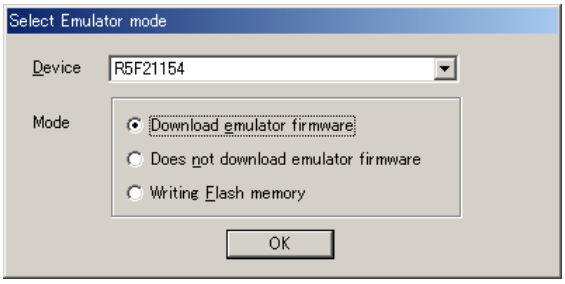


図 3-9 HEW: Select Emulator Mode

「Download emulator firmware」をチェックすると、ファームウエアがターゲットチップ SR8C15CP のフラッシュメモリに書き込まれます。(書き込む領域は以降のウィンドウで選択します。) 一度ファームウエアを書き込んだら、次回の接続ウィザードからは「Does not download emulator firmware」を選んでください。3 つ目の「Writing Flash memory」で機械語ファイルを書き込むか、他のフラッシュメモリ書き込みツールで書き込み、イレースをしないかぎり、一度書き込まれたファームウエアは有効です。

トライアル基板は、E8 より 5V を供給する仕様になっているので、「Power supply is carried out (MAX 300mA)」にチェックを入れて、「5.0V」をチェックしてください。「OK」を押すと、電源が供給され、トライアル基板の電源 LED（燈）が点灯します。

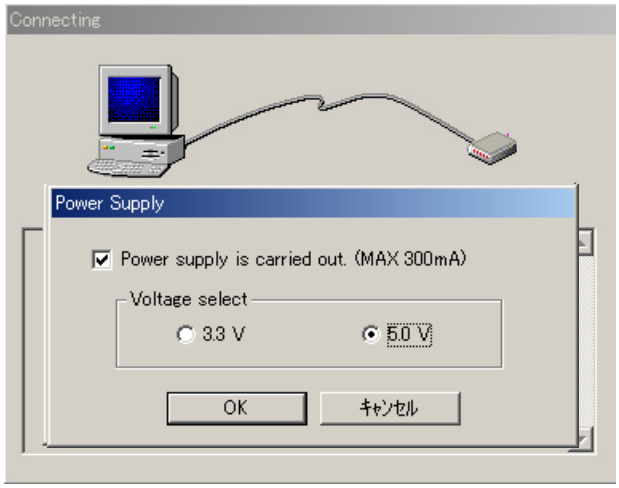


図 3-10 HEW: Power Supply

通信速度はデフォルトの 125000bps のままで「OK」を押してください.

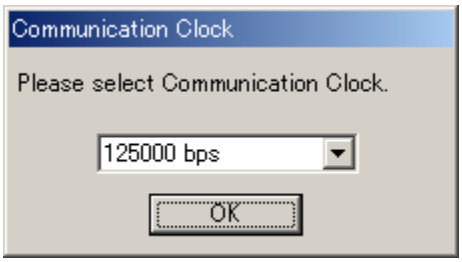


図 3-11 HEW: Communication Clock

ファームウエアを書き込む領域を指定します。SR8C15CP の場合、Data Flash Area をチェックすると 2400h～2BFFh に、User Flash Ares をチェックすると C000h～C7FFh に書き込まれます。サンプルプログラムではプログラムの開始アドレスを C000h に設定しているので、ファームウエアを書き込む領域は Data Flash Area をチェックしてください。
また、2 回目以降接続する場合は、ファームウエアは書き込みませんが、書き込まれている領域を聞いてきますので、Data Flash Area か User Flash Area のどちらに書き込んだかを覚えておいてください。

図 3-12 HEW: Firmware Location

ファームウエアが指定した領域に書き込まれます。

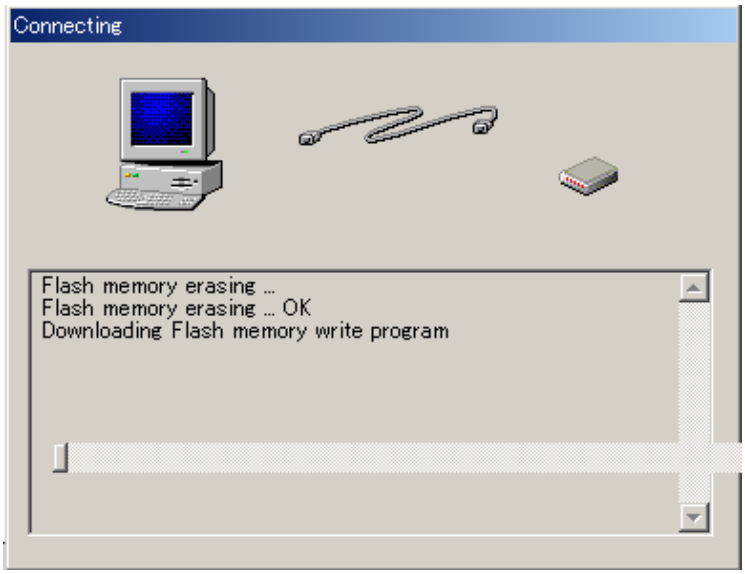


図 3-13 HEW: ファームウエアの書き込み

E8 よりターゲットチップへの書き込み中は、E8 とホスト PC を接続している USB ケーブルや、E8 とターゲットシステムを接続しているケーブルの抜き差しは決してしないでください。また、ホストパーソナルコンピュータ、ターゲットシステムのパワーオフ、リセットは決して行わないでください。

ファームウエアが書き込まれると、E8 を介してホスト PC の HEW とターゲットチップのファームウエアとの通信が開始されます。
ツールバーのデバッグコマンドのアイコンがアクティブになり、エディタウィンドウの左側に実アドレスを表示するスペースができます。（実アドレスはプログラムをダウンロードしたあとに表示されます。）

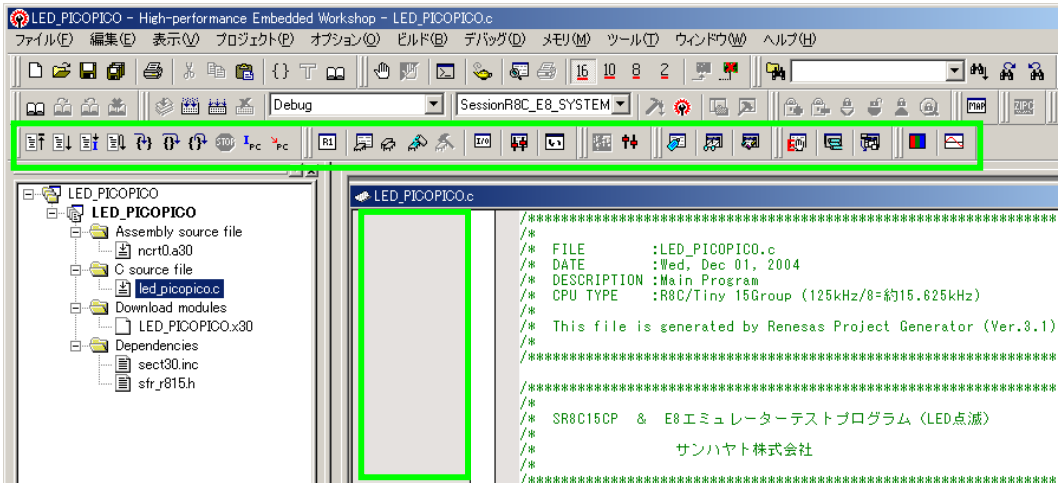


図 3-１４ HEW: E8 と接続後のワークスペース

3-9 プログラムをダウンロードする

アブソリュートモジュールファイルをターゲットチップ SR8C15CP のフラッシュメモリに書き込みます。
「デバッグ」コマンドの「ダウンロード」より C:¥Hew¥Program¥LED_PICOPICO¥led_picopico…を選んでください。（コピーしたフォルダ名先のファイル（.x30）が表示されます。）（図 3-15）
あるいは、ワークスペースウィンドウに表示されている LED_PICOPICO.x30 を選択して右クリックし、ポップアップメニューより「ダウンロード」をクリックしてもダウンロードできます。（図 3-16）

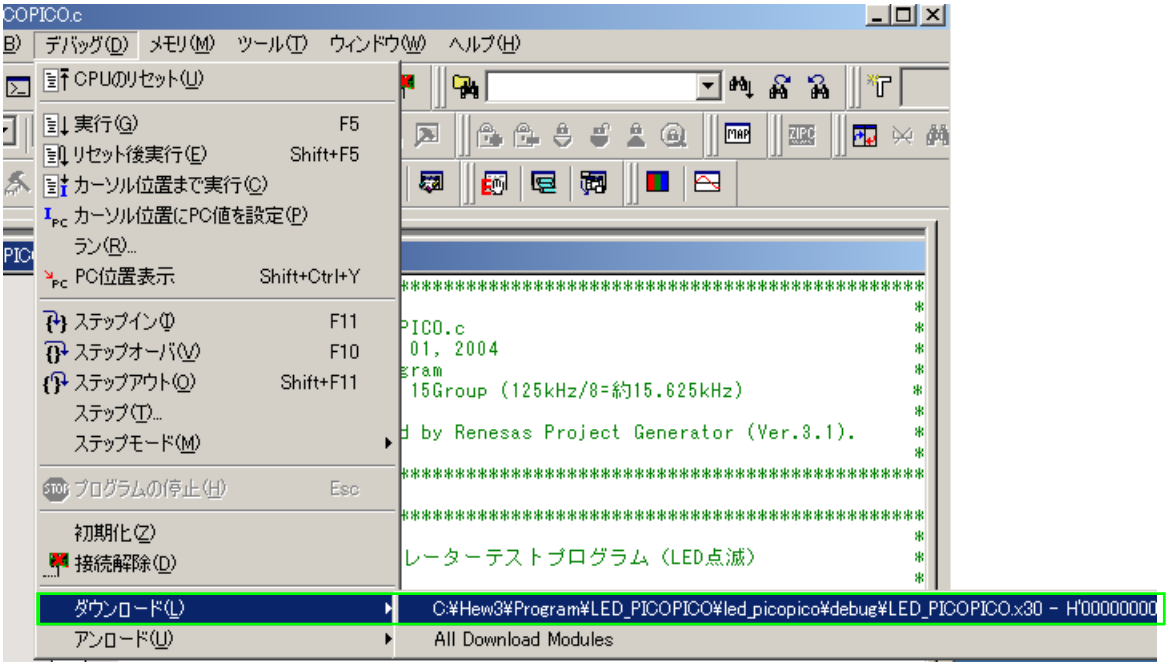


図 3-15 HEW:アブソリュートモジュールファイルのダウンロード 1

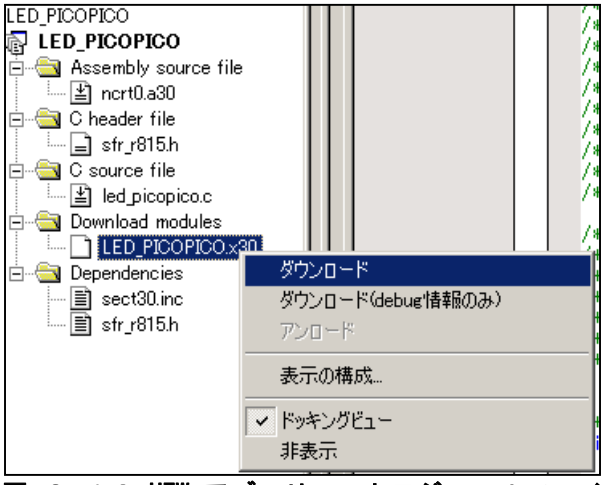


図 3-16 HEW:アブソリュートモジュールファイルのダウンロード 2

E8 よりターゲットチップへの書き込み中は、E8 とホスト PC を接続している USB ケーブルや、E8 とターゲットシステムを接続しているケーブルの抜き差しは決してしないでください。また、ホストパーソナルコンピュータ、ターゲットシステムのパワーオフ、リセットは決して行わないでください。

アブソリュートモジュールファイルが書き込まれると、エディタウィンドウの左側に実アドレスが表示されます。

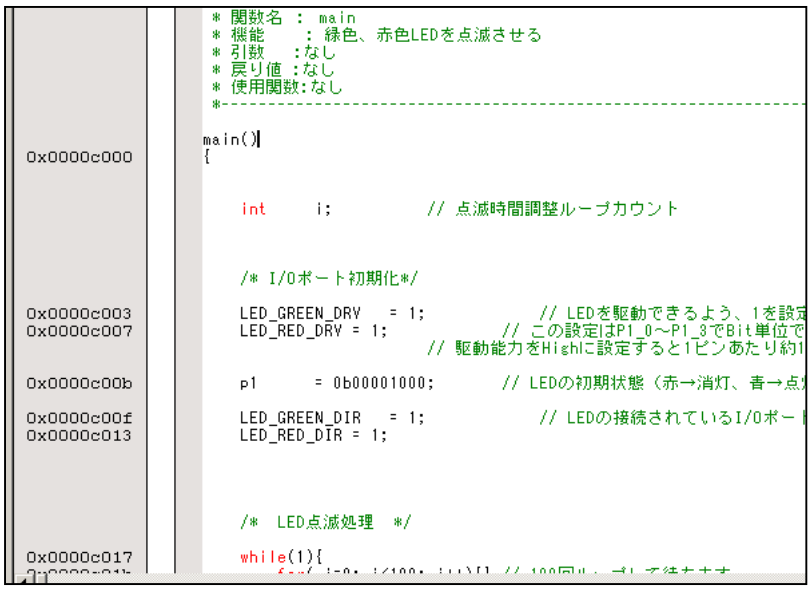


図 3-17 HEW: エディタウィンドウに表示された実アドレス

3-10 プログラムを実行する

「デバッグ」メニューの「実行」か、ツールバーの「実行」アイコンをクリックしてください。プログラムが実行されます。トライアル基板上の LED（緑、赤）が交互に点滅するのを確認してください。

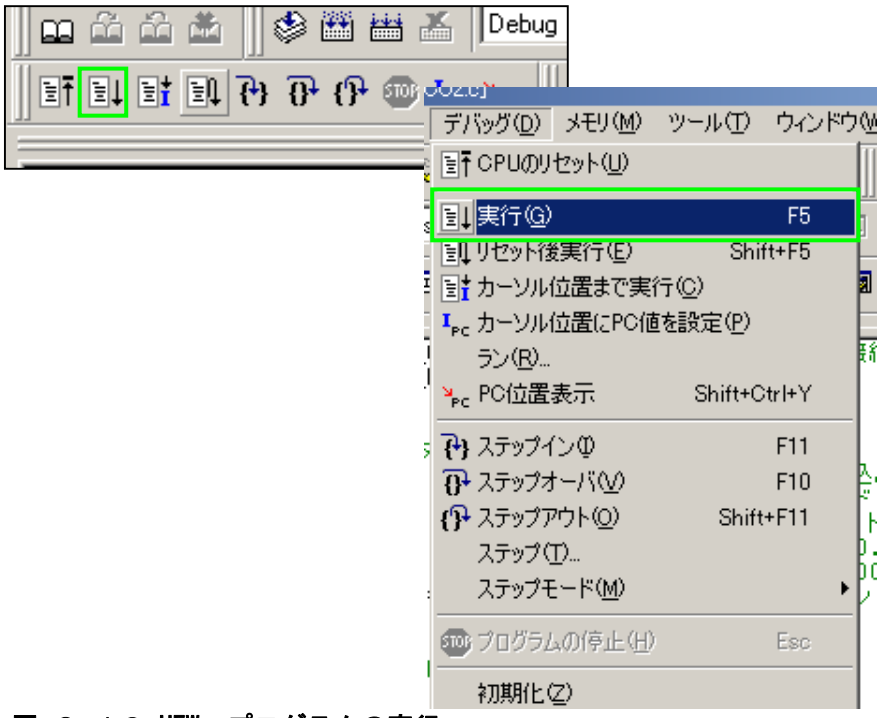


図 3-18 HEW: プログラムの実行

3-11 プログラムを停止する

「デバッグ」メニューの「プログラムの停止」か、ツールバーの「STOP」アイコンをクリックすると、プログラムは停止し、現在の PC（プログラムカウンタ）の位置が黄色の矢印で示されます。

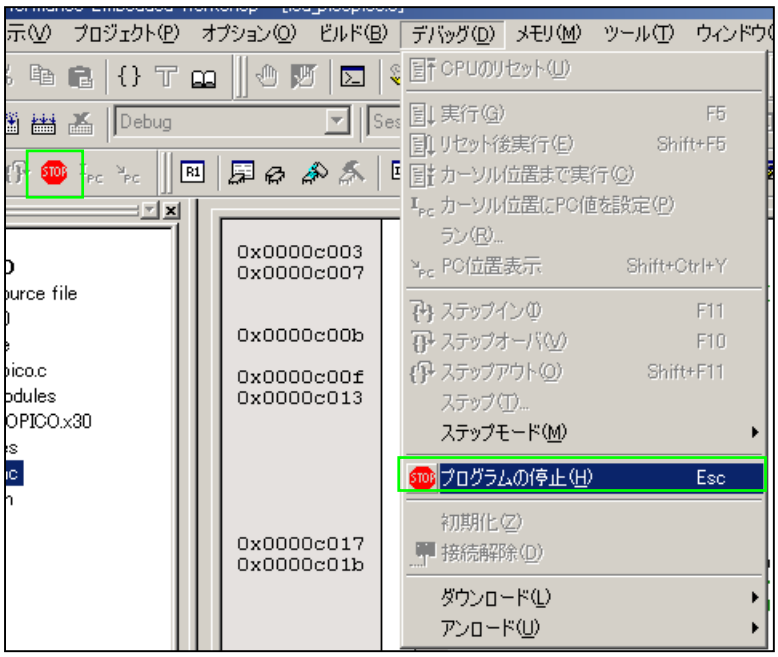


図 3-19 HEW: プログラムの停止

3-12 レジスタ、I/O を表示する

「表示」メニューの「CPU」で、「レジスタ」あるいは「I/O」をクリックしてください。

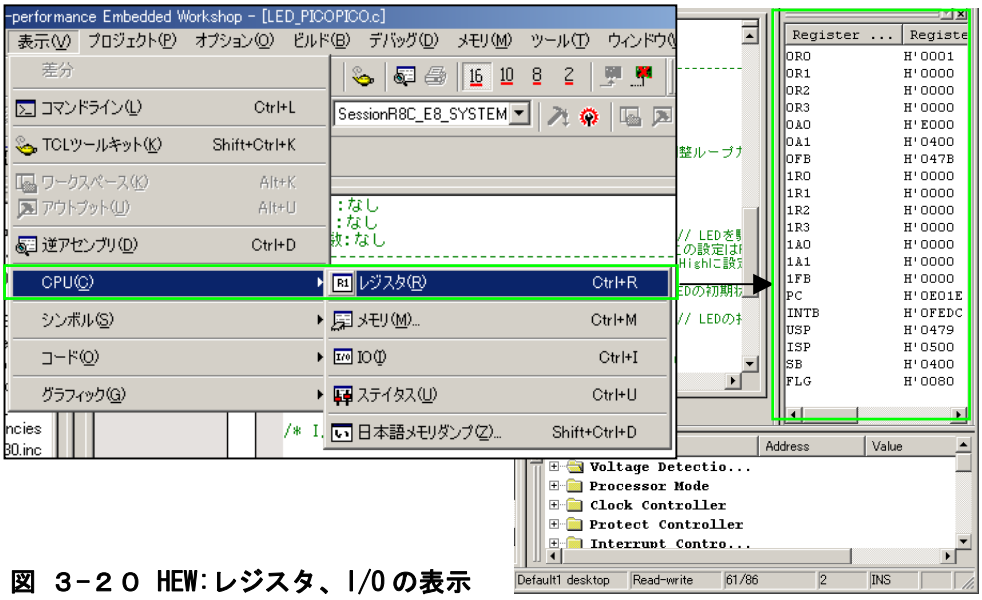


図 3-20 HEW:レジスタ、I/O の表示

3-13 PC の値を変更する

ソースライン上のカーソルの位置のアドレスを、PC（プログラムカウンタ）に設定することができます。プログラム実行を開始したい行にカーソルを置き、「デバッグ」メニューの「カーソル位置の PC 値を設定」をクリックするか、メニューバーの「カーソル位置に PC 設定」アイコンをクリックしてください。PC に設定された行に黄色い矢印が表示されます。

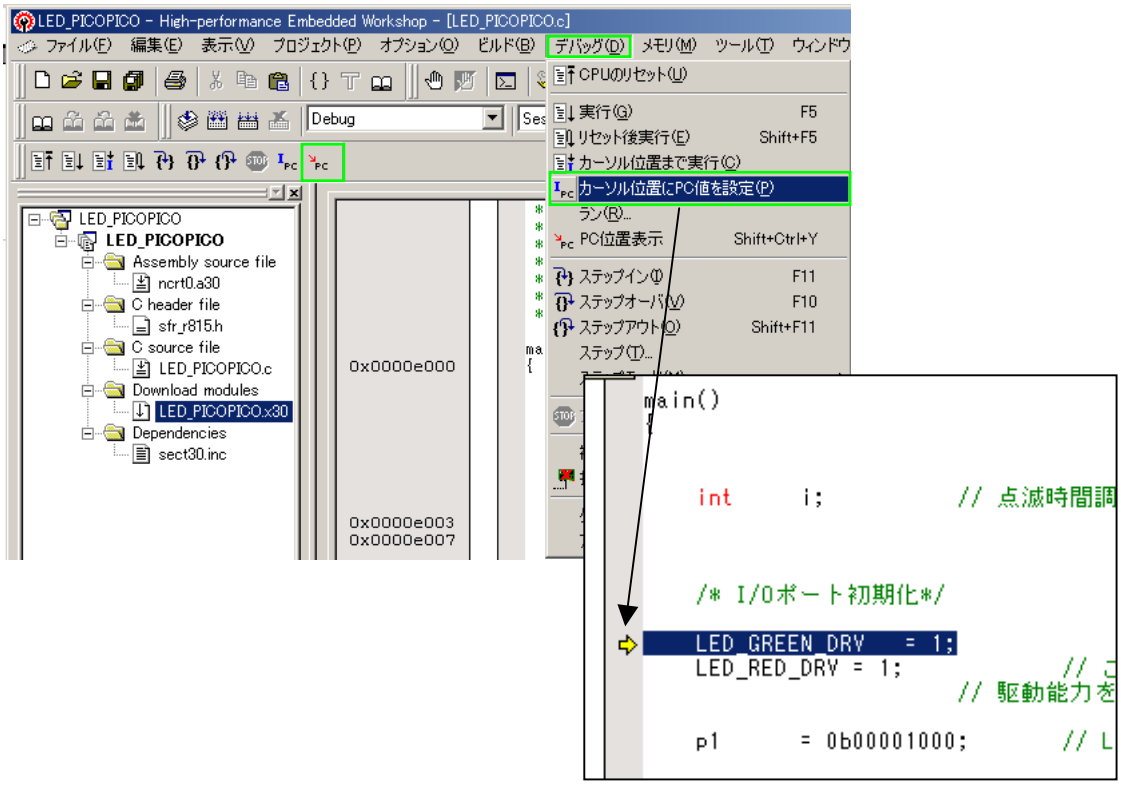


図 3-21 HEW: カーソル位置に PC 値を設定する

3-14 PC ブレークポイントを設定（解除）する

プログラムを停止させたい行にカーソルを置いた状態で、 “F9”キーを押すか、リスト左のグレーの部分をダブルクリックしてください。ブレークポイントが設定された行に赤いマークが付きます。プログラムを実行させると、ブレークポイントが設定された行（命令）のひとつ前の命令までを実行して止まります。
PC ブレークポイントは最大 255 ポイントまで設定できます。

PC ブレークポイントが設定された箇所にカーソルを置いた状態で “F9”キーを押すか、ブレークポイントを直接ダブルクリックすると PC ブレークポイントが解除されます。

※ PC ブレークポイントを設定してプログラムを実行するたびにフラッシュメモリへのプログラムへの書き込みが行われます。書換え可能な回数が減少しますのでご注意ください。

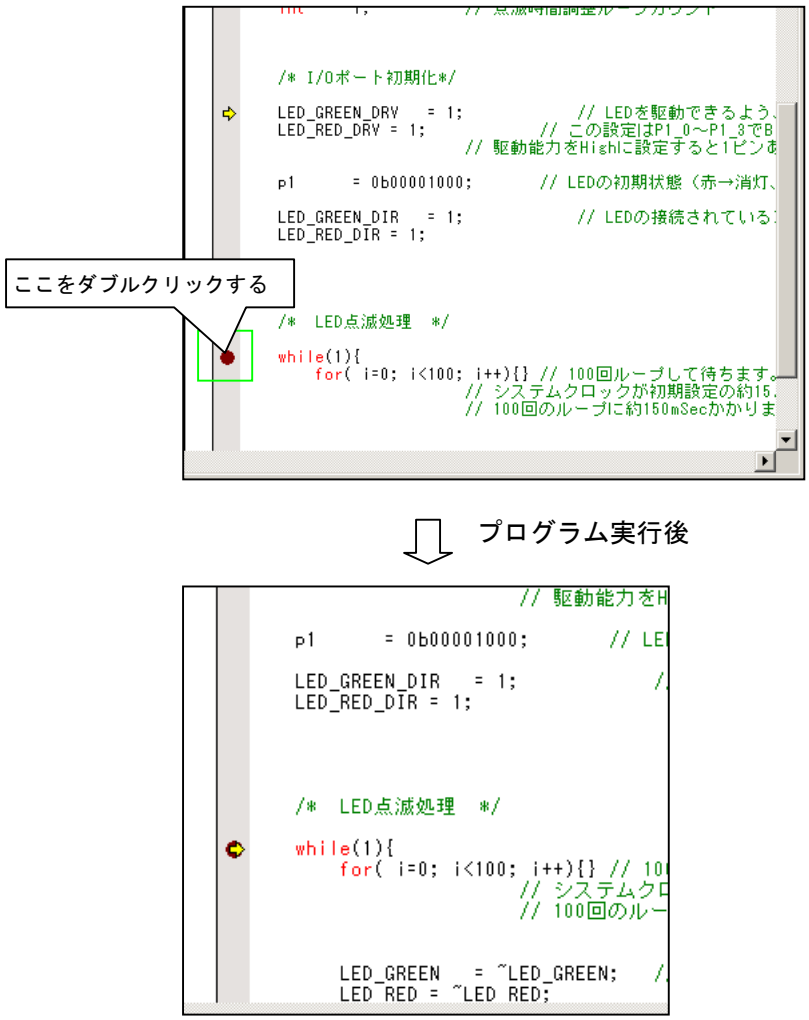


図 3-22 HEW: PC ブレークポイントを設定する

3-15 アドレス一致ブレークポイントを設定（解除）する

プログラムを停止させたい行の左の白い部分を、ダブルクリックをしてください。アドレス一致ブレークポイントが設定された行に青いマークが付きます。プログラムを実行させると、ブレークポイントが設定された行（命令）のひとつ前の命令までを実行して止まります。
設定したアドレスブレークポイントを直接ダブルクリックすると解除されます。

アドレス一致ブレークは、ターゲットチップのアドレス一致割り込み機能を使用しています。SR8C15CP の場合はアドレス一致ブレークを最大 4 ポイントまで設定できます。

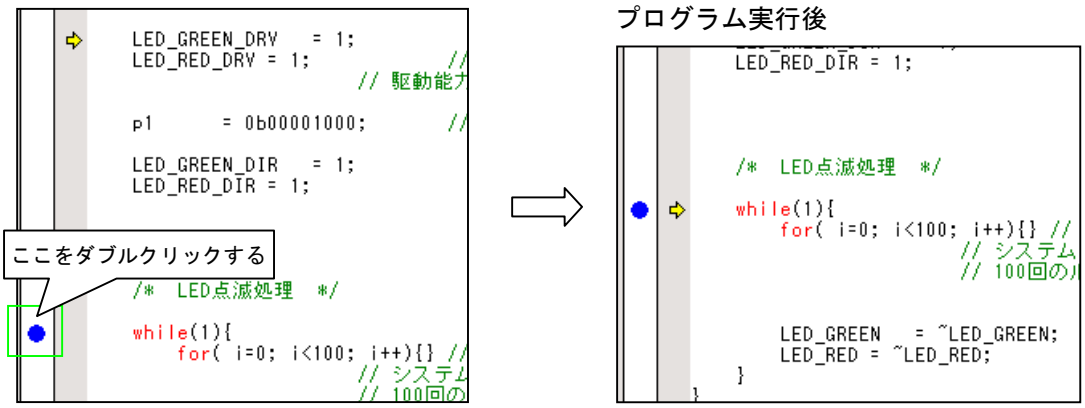


図 3-23 HEW: アドレス一致ブレークポイントの設定

3-16 シングルステップを実行する

シングルステップは 1 行（あるいはアセンブリ言語での 1 命令）を実行します。ソースレベルでの 1 行を実行するか（ソースモード）、アセンブリ言語での 1 命令を実行するか（アセンブリソース）は、「デバッグ」メニューの「ステップモード」で設定します。「自動選択」（デフォルト）に設定すると、現在アクティブになっているエディタウィンドウがソースウィンドウの場合はソースレベルの 1 行、アセンブリウィンドウであれば、アセンブリ言語の 1 命令を実行します。

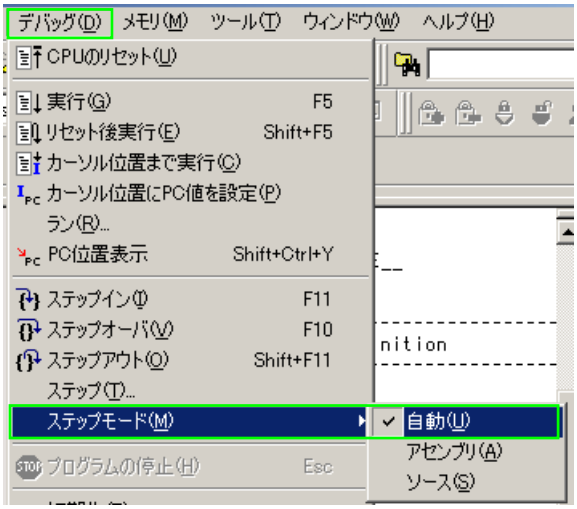


図 3-24 HEW: ステップモードの設定

ステップを実行する行（命令）が関数を呼び出ししている場合の行（命令）の実行の方法として、以下の 2 つから選択できます。

① 「ステップイン」
関数を呼び出す命令のみを実行します。次にステップイン実行した場合、関数の中の次の命令を実行します。

② 「ステップオーバ」
関数を呼び出す命令と関数中の命令を全て（および、さらにその中で呼び出していて実行する可能性のある関数の中の命令全て）実行してから、飛び出し元の、関数を呼び出す命令の次の命令を実行する前で停止します。

関数内の確認したい命令の実行が終了した場合や、誤って関数にステップインした場合、「ステップアウト」を実行することで、関数内の残りの命令を実行せずに呼び出し元の関数に戻る事が出来ます。

「デバッグメニュー」から「ステップイン」、「ステップオーバ」、「ステップアウト」をそれぞれクリックするか、メニューバーの各アイコンをクリックしてください。

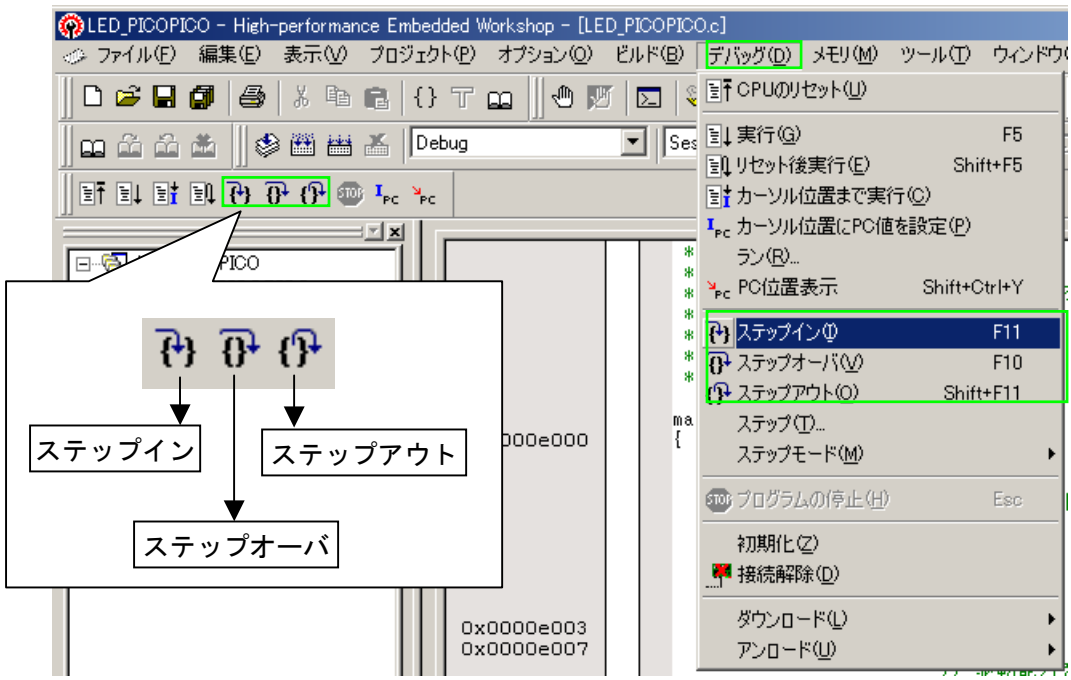


図 3-２５ HEW: ステップ実行

3-17 複数のステップを実行する

「プログラムステップ」ウィンドウに設定することで、一度に複数のステップ実行ができます。ステップの回数、実行する命令（行）の間の時間設定（1～3 秒まで、0.5 秒間隔で設定できます）と、ステップオーバ、ソースレベルステップをそれぞれ選択できます。

「デバッグ」メニューの「ステップ」を選択し、「プログラムステップ」ウィンドウで各設定をしてください。「OK」を押すと設定した複数ステップが実行されます。

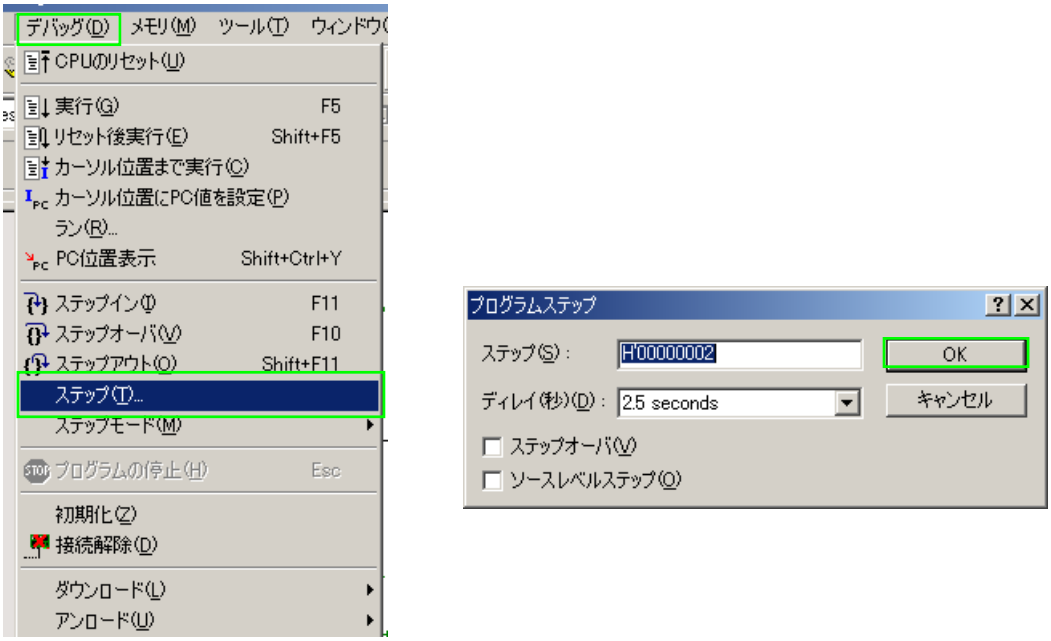


図 3-26 HEW: 複数のステップ実行

3-18 接続を解除する

「デバッグ」メニューの「接続解除」か、メニューバーの「接続解除」アイコンをクリックしてください。

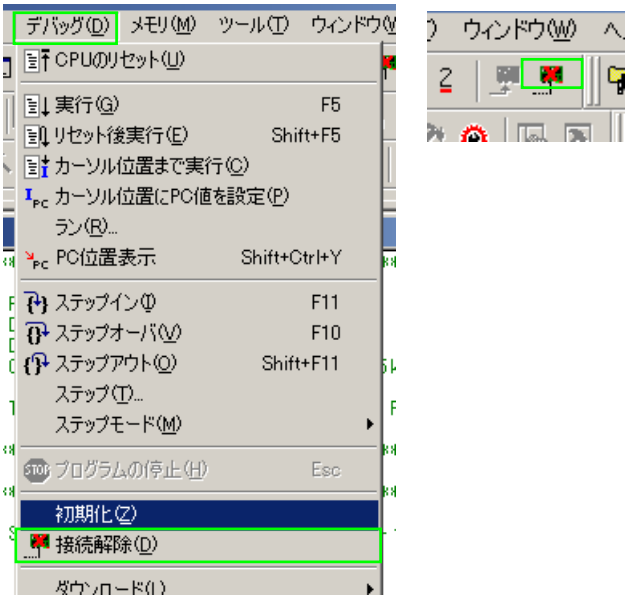


図 3-27 HEW: 接続解除

3-19 接続を再開する

接続を解除したあとに、再び開始するには、「ビルド」メニューの「デバッグ」→「接続」か、メニューバーの「接続」アイコンをクリックしてください。接続ウィザードが開始されます。

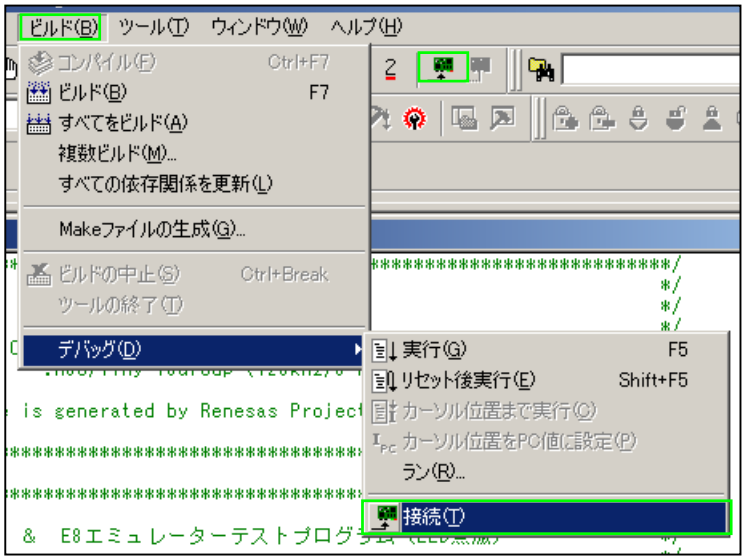


図 3-28 HEW: 接続を再開する

3-20 ワークスペースを閉じる

ワークスペースを一度閉じると、次回開けるときは前回閉じた時点でのセッションから始まります。
サンプルプログラムの場合、DefaultSession の状態でワークスペースを閉じると、次回は DefaultSession で始まります。SessionR8C_E8_SYSTEM のセッションで閉じると、次回は SessionR8C_E8_SYSTEM で始まり、すぐに接続ウィザードが開始されます。

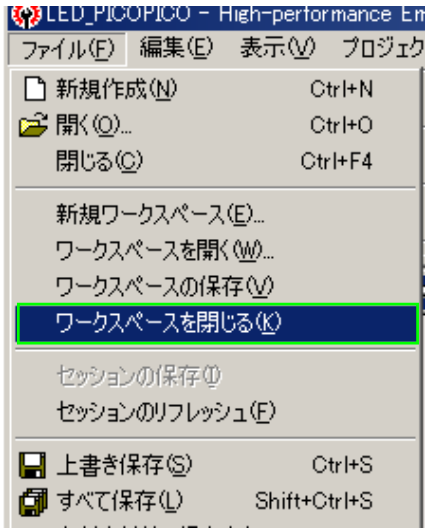


図 3-29 HEW: ワークスペースを閉じる

4 プログラムを作ろう

4-1 新規ワークスペースの作成

まずワークスペースを作成します。「ファイル」メニューの「新規ワークスペース」をクリックしてください。

図 4-1 HEW: 新規ワークスペース

新規プロジェクトワークスペースウィンドウが開きます。
ここでは、C:¥Hew¥Program 以下に LED_PICOPICO2 というワークスペースを作成することを想定しています。(HEW では、ワークスペース、プロジェクト名に英数文字とアンダーバー"_"が使用できます。) 1 つのワークスペースに 1 つのプロジェクトを作成するので、ワークスペースとプロジェクト名は同じ名前にします。「CPU 種別」は「M16C」、ツールチェインは「Renesas R8C Standard」に設定してください。設定したら「OK」をクリックしてください。

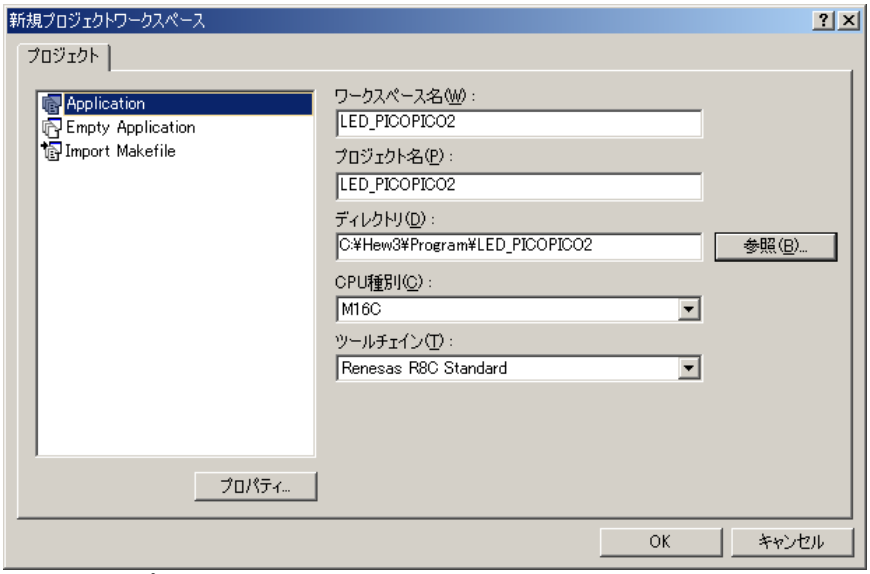


図 4-2 HEW: 新規プロジェクトワークスペース

「Select CPU Toolchain version」ウィンドウが開きます。ここでは設定を変えないで「Next」をクリックしてください。

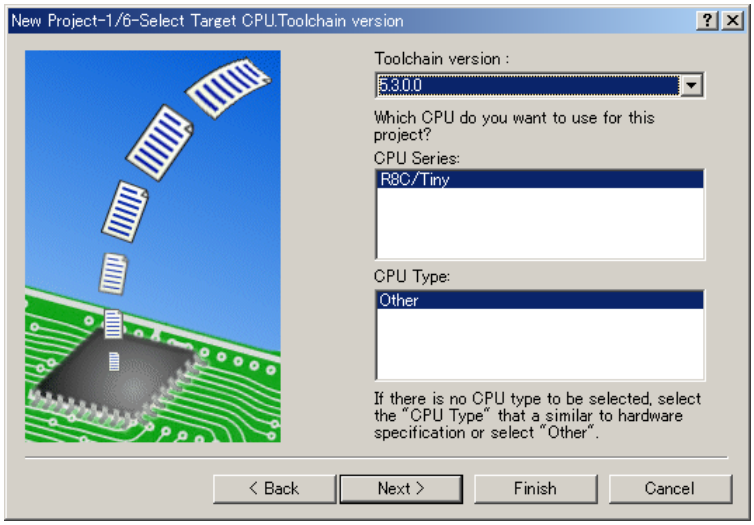

New Project-1/6-Select Target CPU,Toolchain version
Toolchain version :
5.30.0
Which CPU do you want to use for this project?
CPU Series:
R8C/Tiny
CPU Type:
Other
If there is no CPU type to be selected, select the "CPU Type" that a similar to hardware specification or select "Other".
< Back
Next >
Finish
Cancel

図 4-3 HEW: Select CPU Toolchain version

「Select RTOS」ウィンドウでは、「Target type」に「R8C/Tiny」、「RTOS」（リアルタイム OS）は使用しませんので「none」を設定してください。「Startup file type」では、「Default」に設定するとコンパイラ NC8C で提供する標準のスタートアップファイル ncrt0.a30 とセクションファイル sect30.inc がプロジェクトフォルダにコピーされます。自分で独自に作成したファイルを選択する場合は「User」に設定し、「Add」ボタンを押してそれぞれのファイルをファイルエクスプローラで指定します。ここでは NC8C 標準のスタートアップファイルとセクションファイルをコピーすることにします。
設定が終わったら「Next」をクリックしてください。

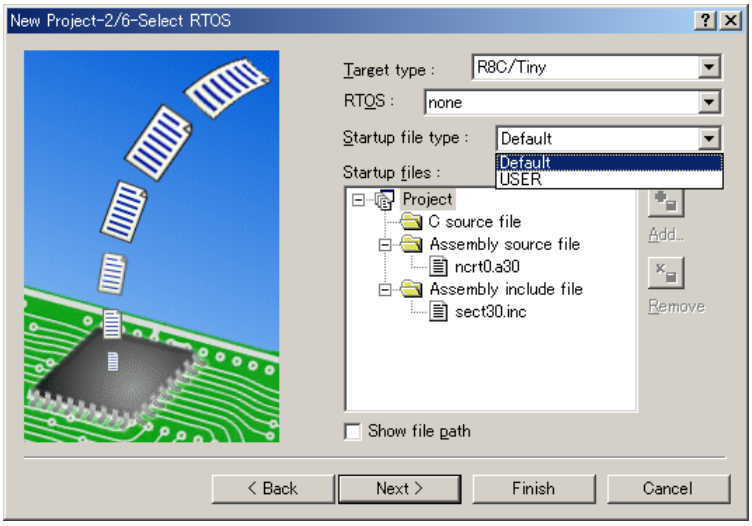


図 4-4 HEW: Select RTOS

「Setting the Contents of Files to be Generated」ウィンドウでは標準 I/O 関数、ヒープ領域の設定を行います。
また「Generate main() Function」では main 関数のファイルを作成するかどうかを指定します。「C source file」に設定すると、プロジェクトフォルダにプロジェクト名と同じ C ソースファイルが作成されます。（この場合は LED_PICOPIC02.C となります。）「None」に設定すると、C ソースファイルは作成されません。
ここでは、標準 I/O 関数、ヒープ領域は使用しません。また、「Generate main() Function」では「C source file」を設定することとします。
設定が終わったら「Next」をクリックしてください。

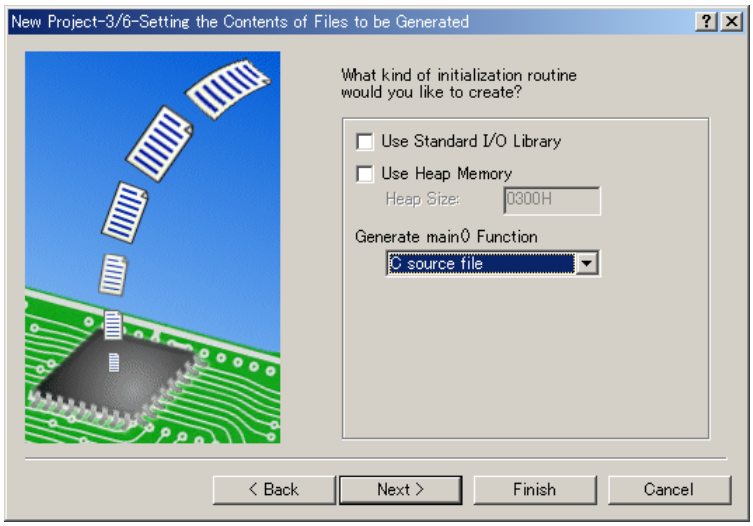


図 4-5 HEW: Setting the Contents of Files to be Generated

次のウィンドウでは、ユーザスタックサイズと割り込みスタックサイズを設定しますが、この設定はスタートアップファイル ncrt0.a30 には反映されません。そのまま「Next」を押してください。

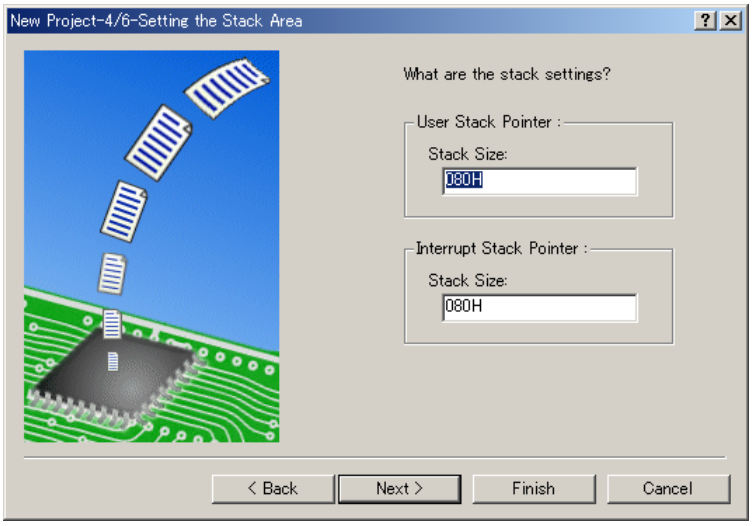


図 4-6 HEW: Setting the Stack Area

「Target System for Debugging」ウィンドウでは、デバッグ時の接続先となるデバッグシステムを指定します。「Targets」には「R8C E8 SYSTEM」をチェックしてください。「Target Type」では「R8C/Tiny」を設定してください。
設定が終わったら「Next」をクリックしてください。

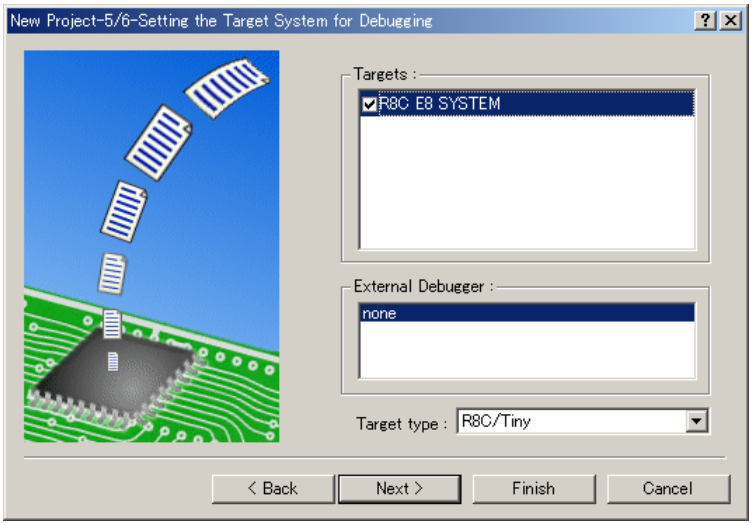


図 4-7 HEW: Target System for Debugging

「Setting the Debugger Options」では、特に設定することはありません。「Next」をクリックしてください。

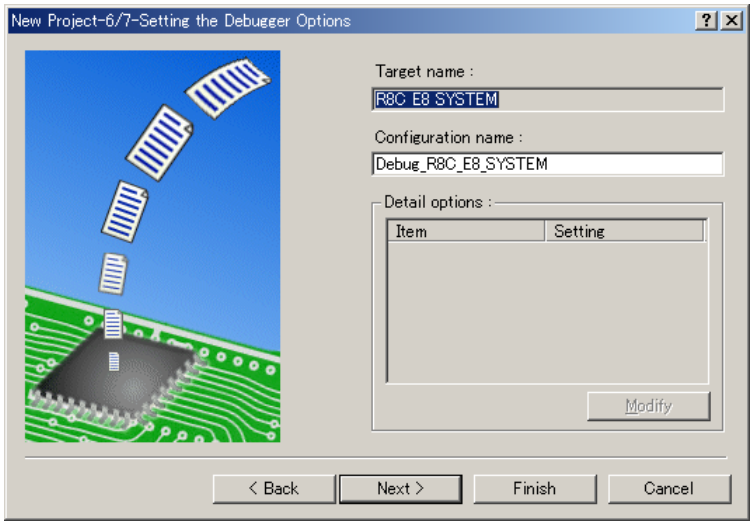


図 4-8 HEW: Setting the Debugger Options

「Changing the File Names to be Created」では、今までの設定で、作成されることになるファイル名を変更することができます。ここでは変更しません。「Next」をクリックしてください。

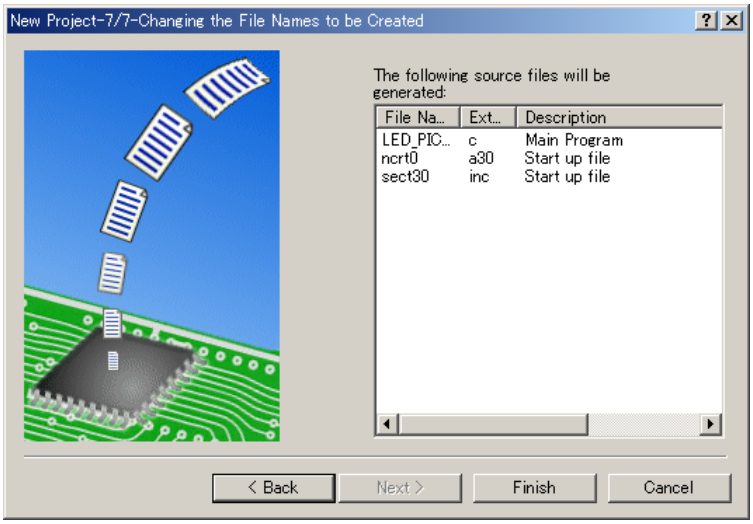


図 4-9 HEW: Changing the File Names to be Created」

設定した項目が表示されます。「Generate Readme.txt as a summary file in the project directory」にチェックを入れておくと（デフォルトでは入っています）、プロジェクトフォルダに「Summary」ウィンドウに表紙されている各設定をテキストファイルにした Readme.txt ファイルを作成します。
「OK」を押してワークスペースの新規作成が終了します。

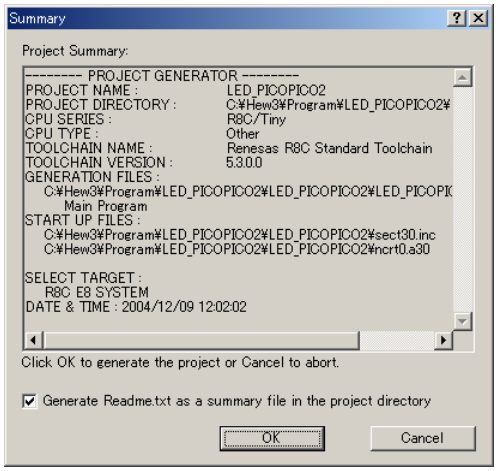


図 4-10 HEW: Summary

4-2 ヘッダファイルを追加する

ヘッダファイル　sfr_r815.h をプロジェクトに追加します。
まず、Windows のファイルエクスプローラで、サンプルプログラムの LED_PICOPICO プロジェクトフォルダより、LED_PICOPICO2 プロジェクトフォルダへ sfr_r815.c をコピーしてください。

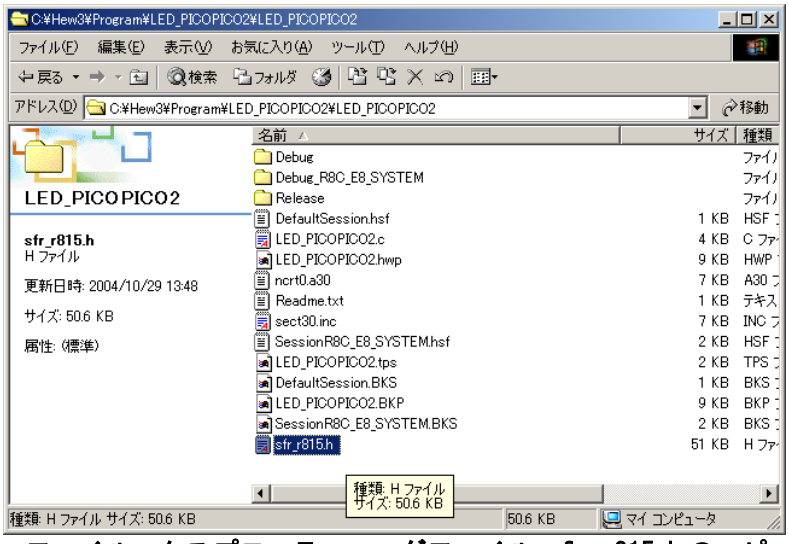


図 4-11 Windows ファイルエクスプローラ: ヘッダファイル sfr_r815.h のコピー

次に HEW でプロジェクトに sfr_r815.c ファイルを追加（登録）します。
「プロジェクト」メニューの「ファイルの追加」をクリックしてください。
「ファイルの追加」ウィンドウで、プロジェクトフォルダ（LED_PICOPICO2）を指定し、必ず「ファイルの種類」を「C header file (*.H)」に設定して、sfr_r815.h を選択してください。

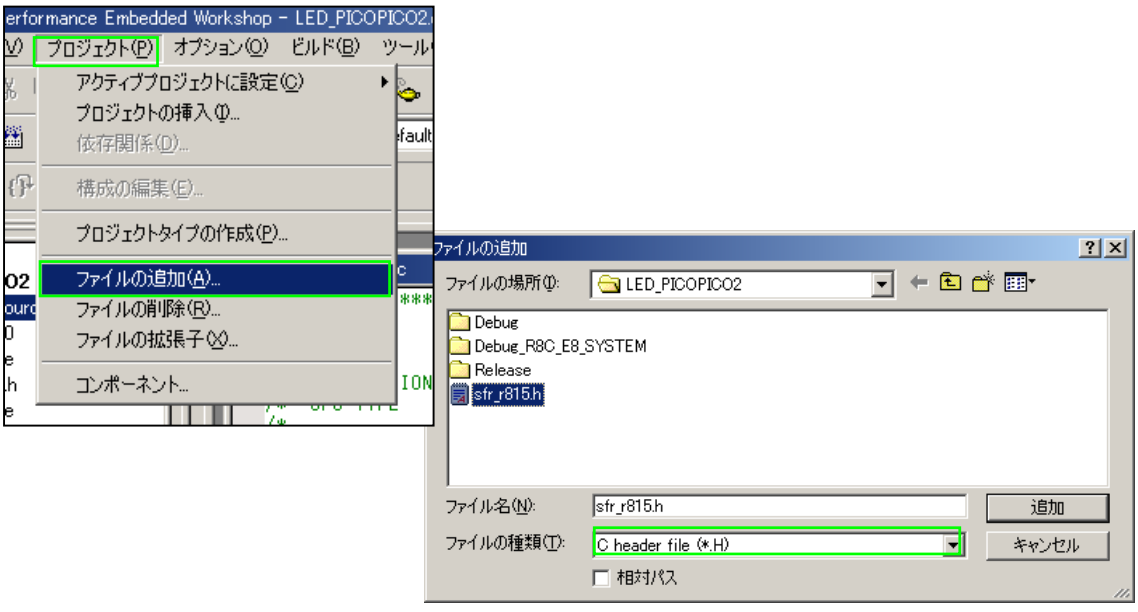


図 4-12 HEW: ファイルの追加

4-3 コーディングする

プロジェクトスペースの「LED_PICOPICO2.c」をダブルクリックして、エディタウィンドウを開き、コーディングしてください。コーディングが終了したらメニューバーの「ファイルの保存」アイコンをクリックするか、「Ctrl＋S」キーを押してファイルを保存してください。

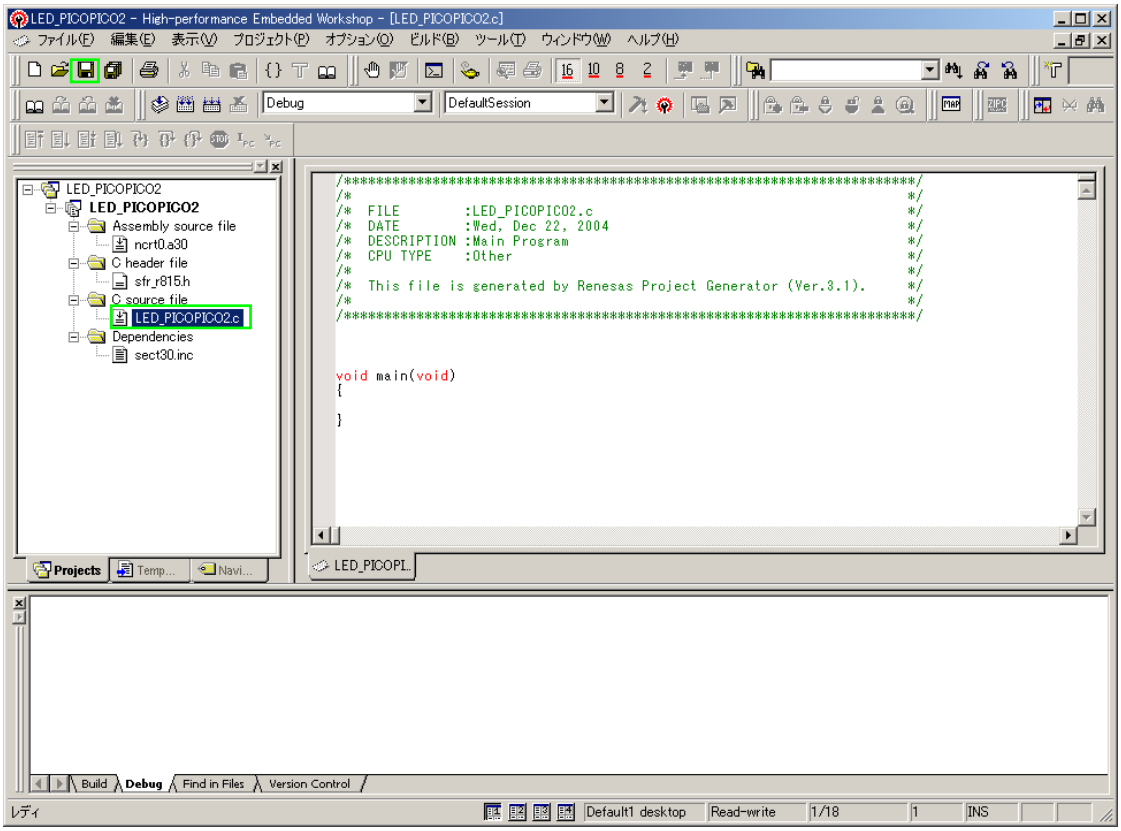


図 4-13 HEW:コーディング

以下にサンプルプログラム「LED_PICOPICO2.c」の参考リストを示します。タイマ X の割り込みフラグをポーリングで検知することで 0.1 秒間隔の LED 点滅を実現しています。

```
/***********************************************************************/
/*                                                                     */
/*  FILE        :LED_PICOPICO2.c                                       */
/*  DATE        :Wed, Dec 01, 2004                                     */
/*  DESCRIPTION :Main Program                                          */
/*  CPU TYPE    :R8C/Tiny 15Group (125kHz/8=約15.625kHz)               */
/*                                                                     */
/*  This file is generated by Renesas Project Generator (Ver.3.1).     */
/*                                                                     */
/***********************************************************************/

/*****************************************************************************************************/
/*                                                                                                   */
/*  SR8C15CP　&　E8エミュレーターテストプログラム（LED点滅）                                          */
/*                                                                                                   */
/*                          サンハヤト株式会社                                                       */
/*                                                                                                   */
/*****************************************************************************************************/


/* インクルードファイル */
#include "sfr_r815.h"                           //SR8C15CPのI/Oレジスタが定義してあるSFRヘッダ（C言語用）

/* プロトタイプ宣言 */

/*I/Oの定義*/

/* 使用するI/Oレジスタの定義
 * LEDは高電流駆動能力が選択できるPORT１に配線されている
 */

#define     LED_GREEN           p1_1        // 緑LEDはP1_1（Pin17）に配線されている
#define     LED_RED             p1_3        //赤LEDはP1_3（Pin14）に配線されている

#define     LED_GREEN_DIR       pd1_1       // 緑LEDのポートの方向レジスタ
#define     LED_RED_DIR         pd1_3       // 緑LEDのポートの方向レジスタ

#define     LED_GREEN_DRV       drr1        // 使用するI/Oポートの駆動能力を設定するbit
#define     LED_RED_DRV         drr3        //1を設定すると駆動能力Highになり、LEDが駆動できる
```

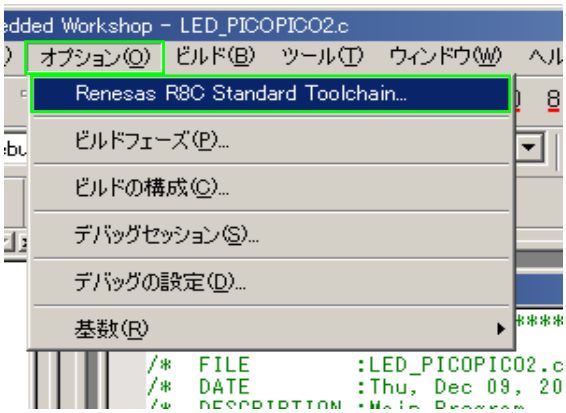
リスト 4-1 LED_PICOPICO2.c ヘッダ、各定義

```
/*-------------------------------------------------------------------------
 * ID       :なし
 * 関数名   :  main
 * 機能          : 緑色、赤色LEDを点滅させる(約0.1秒)
 * 引数     :なし
 * 戻り値 :なし
 * 使用関数:なし
 *-------------------------------------------------------------------------*/

main()
{



        /* I/Oポート初期化*/
        LED_GREEN_DRV       = 1;        // LEDを駆動できるよう、1を設定する
        LED_RED_DRV         = 1;        //  この設定はP1_0～P1_3でBit単位で設定できる
                                        //  駆動能力をHighに設定する
        p1          = 0b00001000;       // LEDの初期状態（赤→消灯、青→点灯）

        LED_GREEN_DIR       = 1;        // LEDの接続されているI/Oポートを出力に設定する
        LED_RED_DIR         = 1;


        /*  タイマX初期化*/
        txic = 0b00000000;              //タイマX割り込み制御レジスタ→タイマX割り込み禁止
        txmr = 0b00000000;              //タイマxモードレジスタ→タイマモード、カウント停止
        tcss = 0b00000000;              //タイマカウントリソース設定レジスタ→f1(125KHz)に設定
        prex = 125;                     //1秒/125KHz=0.000008(1周期が0.000008秒)
        tx = 100;                       //0.1秒/0.000008秒=12500カウントを125と100に分けて設定
        txs = 1;                        //タイマXカウント開始


        /*   LED点滅処理   */

        while(1){
                if(ir_txic==1){         //タイマXのアンダフローでタイマX割り込み要求が1になる
                        LED_GREEN = ~LED_GREEN;         //  ポートレジスタの値を反転させる
                        LED_RED   = ~LED_RED;
                        ir_txic = 0;                    //次の割り込み要求検知のため、クリアする
                        }
        }
}
```

リスト 4-2 LED_PICOPICO2.c main 関数

4-4 オプションの設定をする

ソースレベルデバッグに必要な-G オプション（コンパイラオプション:デバッグ情報をファイルに出力する）を設定します。この設定はビルドコンフィギュレーション「Debug」に登録します。
「オプション」メニューの「Renesas R8C Standard Toolchain」をクリックしてください。
「Renesas R8C Standard Toolchain」ウィンドウが開きます。
「Configuration」では、かならず登録するコンフィギュレーションに「Debug」を指定し、対象となるプロジェクトは「LED_PICOPICO2」にしてください。C コンパイラの「Category」を「Object」にし、「Debug options」の「[-g] Output Debugging information…」にチェックを入れてください。



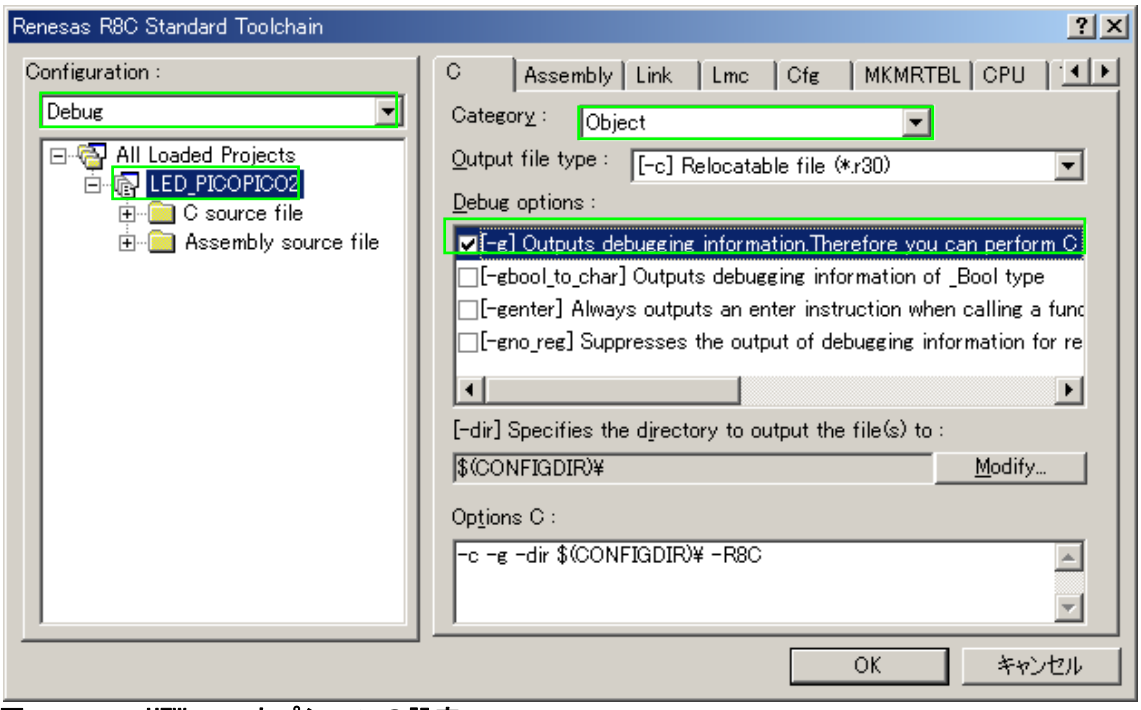


図 4-14 HEW: -g オプションの設定

4-5 ダウンロード後にリセットされるように設定する

アブソリュートモジュールをダウンロードする度に、ターゲットチップがリセットされ PC（プログラムカウンタ）にスタートアドレスが設定されるようにします。この設定は E8 と接続するデバッグセッション「SessionR8C_E8_SYSTEM」に登録します。「オプション」メニューの「デバッグの設定」をクリックしてください。かならず登録するデバッグセッションに「SessionR8C_E8_SYSTEM」を指定し、「ダウンロード後に CPU をリセットする」にチェックを入れてください。

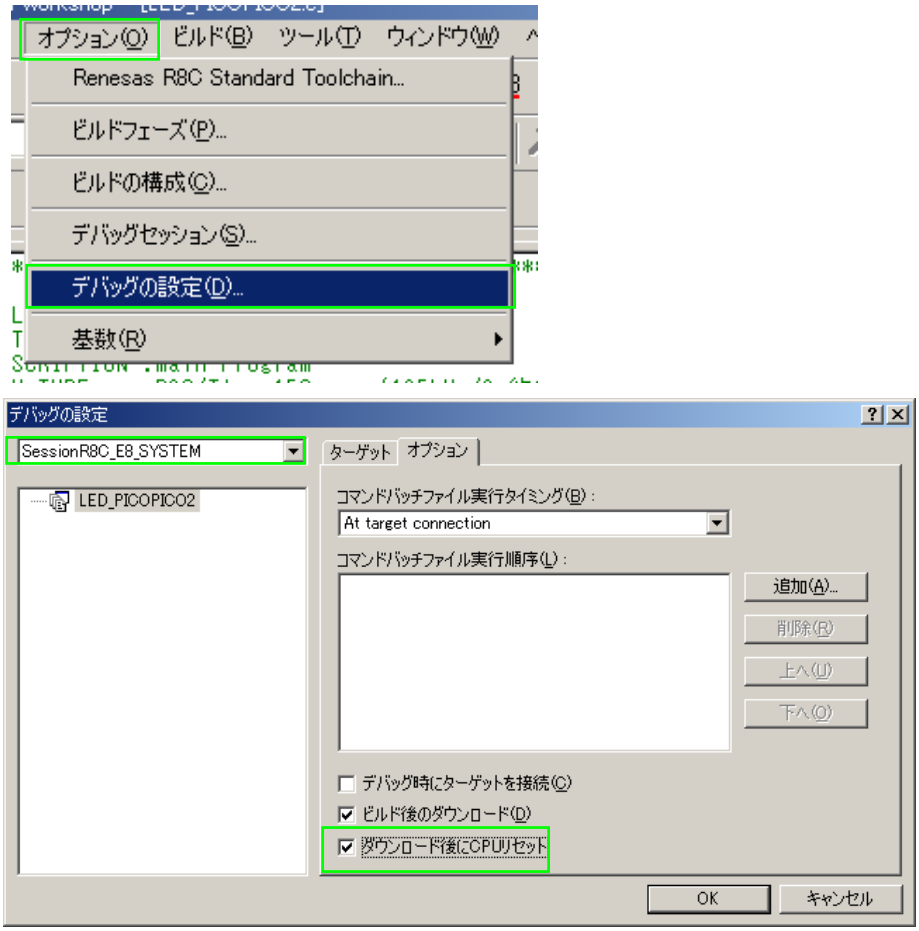


図 4-15 HEW: ダウンロード後のリセットの設定

ここまでの設定で、サンプルプログラム「LED_PICOPICO」と同じ内容の設定となります。
プロジェクトを一度保存してください。

あとは、「3. サンプルプログラムを動かそう」を参照して、ビルド、E8 との接続、プログラムのダウンロード/実行を行ってみてください。

5 注意事項

R8C 専用無償コンパイラ NC8C の機能制限、また E8 を使用して評価を行う際の注意事項を示します。

5-1 NC8C の機能制限

ルネサステクノロジ社より提供される無償版 R8C/Tiny シリーズコンパイラ NC8C（E8 同梱の CD にも収められています）には、正規版コンパイラ(NC30WA)に対して以下のような機能の制限があります。

表 5-1 NE8C の機能比較表

	プロフェッショナル版コンパイラ（最新版） M3T-NC30WA	専用無償版コンパイラ M3T-NC8C
バージョン	V.5.30 Release 1	プロフェッショナル版 V.5.30 Release 1 相当
ユーティリティツール	あり（MapViewer、STKViewer、lib30）	なし
プログラムリンクサイズ制限	なし	あり（64K バイト）
マニュアル	あり	あり
使用期間	無制限	無制限
対応デバイス	M16C/60、30、Tiny、20、10、シリーズ R8C/Tiny シリーズ	R8C/Tiny シリーズ
サポート	無償バージョンアップ ツール技術サポートあり	なし
動作確認済み OS	IBM PC/AT 互換機 （Windows98/Me/NT4.0/2000/XP, 日本語 Turbolinux 7 Workstation) HP9000 シリーズ 700（HP-UX 10.2x） SPARCstation（Solaris 2.5）	IBM PC/AT 互換機 （Windows98/Me/NT4.0/2000/XP）

５-２ E8 を使用する際の制限事項

E8 は通信開始時にフラッシュメモリのデータ領域（2400h～27FFh）か、プログラム領域（C000h～C7FFh）のどちらかにファームウエアを書き込みます。どちらの領域に書き込むかはユーザが選択できます。ユーザプログラムがファームウエアを書き込む領域に重ならないようにしてください。

５-３ E8 を使用する際の注意事項

E8 を使用する際の注意事項については、E8 デバッギングエミュレータに同梱の CD に収められている、ルネサステクノロジ社発行の「E8 エミュレータ　ユーザーズマニュアル　別冊 R8C/14, R8C/15, R8C/16, R8C/17 接続時の注意事項」の「4. E8 エミュレータと各 MCU の相違点」を参照してください。

デバッグは、ホスト PC 上の HEW が E8 を介してファームウエアと通信を行う事で実現しています。デバッグ中は、HEW からのコマンドに従って、ターゲットチップがファームウエアとユーザプログラムを交互に実行していることになります。この原理をふまえ、注意事項に留意しておくことで HEW および E8 を使用した開発環境を最大限に活用することができます。

改訂履歴

Rev.	発行日	ページ	改訂内容
1.00	2004/12/15	-	初版発行

SR8C15CP マイコン IC
スタートアップガイド　E8 エミュレータ導入編 1

発行日 2004 年 12 月 15 日　Rev1.00
発　行 サンハヤト株式会社
〒170-0005 東京都豊島区南大塚 3 丁目 40 番 1 号
編　集 サンハヤト株式会社 研究開発部
©2004 Sunhayato Corp. All rights reserved. Printed in Japan.
SG043101